AFRL-IF-RS-TR-2005-188
Final Technical Report
May 2005

DYNAMIC CONTROL AND FORMAL MODELS OF MULTI-AGENT INTERACTIONS AND BEHAVIORS

BAE Systems Advanced Information Technologies

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. K542

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2005-188 has been reviewed and is approved for publication

APPROVED: /s/

JAMES M. NAGY
Project Engineer

FOR THE DIRECTOR: /s/

JOSEPH CAMERA, Chief
Information & Intelligence Exploitation Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	MAY 2005	Final Jun 00 – Jan 05

4. TITLE AND SUBTITLE
DYNAMIC CONTROL AND FORMAL MODELS OF MULTI-AGENT INTERACTIONS AND BEHAVIORS

5. FUNDING NUMBERS
C - F30602-00-C-0182
PE - 62301E
PR - TASK
TA - 00
WU - 01

6. AUTHOR(S)
Larry Roszman, Derek Armstrong,
Aram Khalali and Gwen Hickling

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
BAE Systems Advanced Information Technologies
3865 Wilson Road, Suite 600
Arlington Virginia 22203

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFED
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2005-188

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: James M. Nagy/IFED/(315) 330-3173/ James.Nagy@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 Words)*
New Multi-Agent System (MAS) approaches to complex DoD problems hold the promise of previously unrealized levels of autonomy, adaptability, and flexibility of agent-controlled systems. These systems will provide essential capabilities in command and control, surveillance, automated targeting and weapons delivery, and biochem monitoring.
BAE SYSTEMS Advanced Information Technologies' work focused on three areas. First was the development of the Open Experimentation Framework to facilitate research, evaluation, and characterization of the emerging science of Multi-Agent Systems. Second was the design and facilitation of a project-wide demonstration in which all Principal Investigators participate. Third was our theoretical research into cooperative and adaptive methods for multi-agent systems to service asynchronously appearing popup tasks.

14. SUBJECT TERMS
Multi-Agent Systems, Agent-Based Computing, Cooperation, Adaptation, Scientific And Mathematical Foundations, Autonomous Operation, UAV

15. NUMBER OF PAGES
51

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

The objective of BAE SYSTEMS Advanced Information Technologies' (AIT) effort, performed under the DARPA Taskable Agent Software Kit (TASK) program, was to extend the current scientific and mathematical foundations of agent-based computing by adding rigor to the engineering of agent-based systems and tools. To meet this objective, AIT performed the following tasks:

- **The Open Exploration Framework (OEF):** the OEF was a well-defined abstract problem context in which researchers could apply dissimilar approaches to a standard problem set and compare results on a standard basis. AIT defined the common problem set, the characteristics and parameters of the OEF, and bases for results comparison. In addition, AIT facilitated discussions about the OEF among the TASK Principal Investigators (PIs) and incorporated into the OEF consensus new and additional concepts that arose during these discussions.

- **The TASK OEF Website**: AIT maintained a website with links to the TASK Principal Investigator's TASK websites, with presentations made by the TASK Principal Investigator's at the scheduled PI meetings, with papers provided by the Principal Investigator's, with links to other websites of interest, and with copies of other papers of interest to TASK and the OEF.

- **Multi-Agent System (MAS) Research:** AIT investigated approaches to the design and control of intelligent cooperation and adaptation within systems of autonomous, communication intelligent agents that perform tasks in rapidly changing environments. During this period the research effort focused on techniques for agents operating within a multi-agent system to coordinate on minimizing the time a task waits between when it is available for service and when it is serviced by the multi-agent system.

- **The AIT Testbed for Taskable Agent Systems (TTAS):** AIT developed TTAS, a highly instrumented, simulation environment in which the same, repeatable experiments can be performed upon adaptive systems of cooperating, autonomous, intelligent agents. Because of TTAS's standard application programming interfaces (APIs), messaging services, and repeatable environment, researchers can compare similar measurements made on different MAS implementations under the same experimental conditions and draw meaningful conclusions about comparisons of the different MAS implementations.

- **The TASK Project-wide Demonstration:** AIT designed, organized, and facilitated a project-wide demonstration for the DARPA TASK program for all PI organizations. This demonstration took place, August 4, 2004 in the ballroom of the Hotel Washington, Washington, DC. Nearly thirty people from various governmental and commercial organizations attended.

- **The AIT Demonstrations for the TASK Project-wide Demonstration:** AIT provided two demonstrations of its research into multi-agent systems. One was a software demonstration of the adaptation and cooperation algorithms that resulted from AIT's MAS research. The second was a hardware demonstration of a system of autonomous robots and MICA2 Motes

cooperating to locate and survey acoustic targets. The algorithms developed in AIT's MAS research were implemented on the robots for the hardware demonstration.

1.1 OBJECTIVES AND ACCOMPLISHMENTS

AIT performed six tasks. The first, the OEF provided a common research context for all TASK Principal Investigators. In the second task, AIT maintained a website for TASK and the TASK OEF. In the third, AIT performed research into several approaches to adaptation and coordination in MASs. In the fourth, AIT continued development of its general testbed simulator TTAS in which experiments can be performed on different and mixed MAS implementations. For the fifth, AIT facilitated a project-wide demonstration for all TASK principal investigators. In the sixth, AIT developed two AIT demonstrations that featured different aspects of AIT's MAS research and presented them at the project-wide demonstration.

- **The OEF:** The TASK Principal Investigators used the OEF, described within Section 2 in detail, and the UAV surveillance and reconnaissance problems that AIT defined as a focus for the application of much of their research. Several groups used the OEF problems as a basis for collaboration, and others exchanged related data and software. Discussions about the OEF and metrics associated with the OEF engendered lively and thoughtful discussion during teleconferences and at meetings. AIT crafted metrics for the OEF based upon the various discussions, but the project ended before all of the Principal Investigators agreed that these metrics were sufficient and complete. Overall, the OEF was a success.

- **The TASK OEF Website:** The Principal Investigators, the DARPA Program Manager, and DARPA support personnel commented that they used the website to become acquainted with the details of other Principal Investigators work and, in general, found the website quite useful. AIT provided a copy of the entire website to the DARPA Program Manager through the DARPA support personnel at the end of the TASK project.

- **MAS Research:** AIT examined several possible approaches to collaboration and adaptation in multi-agent systems and chose to investigate optimization-based techniques for an agent to select specific tasks for servicing and determining the optimal order in which the agent should service the tasks. Secondary collaboration and adaptation activities rely upon inter-agent messages for one agent to inform others that it serviced a task and to schedule a cross-mission task servicing with another agent. Section 3 contains an extensive description of the algorithmic approach and the numerical results of simulations.

- **TTAS:** AIT provided a complete description of the design and operation of the Testbed for Taskable Agent Systems in an earlier report (*Dynamic Control and Formal Models of Multi-Agent Interaction and Behaviors: Final Technical Report*, Nov. 27, 2002). During the period covered in the current report, AIT performed maintenance and minor enhancements to support AIT's MAS research effort. AIT found that TTAS is very good for both the rapid survey of MAS implementations and for in-depth experimentation with a specific MAS implementation under diverse conditions.

- **TASK Project-wide Demonstration:** The project-wide TASK demonstration, held August 4, 2004, at the Hotel Washington, Washington D.C., was an opportunity for government,

industrial, and academic researchers and developers to sample the results of the TASK project. The demonstration was a modest success with all TASK Principal Investigators presenting demonstrations and approximately thirty people from outside the TASK project attending. Section 4 contains a description of the basis of the demonstration, which was an extension of the OEF. Those who attended were quite enthusiastic about the demonstrations. With additional follow-up on the initial invitations, it is likely many more people would have seen the demonstration and had similar opinions.

- **The AIT Demonstration:** AIT presented a software-based demonstration that used TTAS and presented the details and the results of AIT's research into optimization-based adaptation and collaborations algorithms for a multi-agent system servicing asynchronously arriving popup tasks. In addition, AIT presented a hardware-based demonstration of robots and sensor networks executing the algorithms crafted during AIT's MAS research efforts. The robots acted as UAV stand-ins and had a fully complement of video, laser, and sonar sensing, on-board computing, and communications with both robots and the nodes of the sensor network. The hardware demonstration, described in Section 5, served as an extension to AIT's basic MAS research. In this extension, AIT examined the compromises and complexities necessary to bring theoretical algorithms to real hardware. Surprisingly, few compromises occurred. By using TTAS, a simulated experimental environment that used an abstract but realistic model of UAVs, AIT researchers had encountered almost all such difficulties during development of the algorithms.

2 The Open Experimentation Framework (OEF)

The first purpose of the OEF was to provide a common military context in which TASK Principal Investigators could perform cooperative research or, while using quite different techniques, produce results they could compare or use complementarily. The second purpose or the OEF was to focus the Principal Investigators' efforts on important and difficult problems common to all multi-agent systems. The meta-problem that AIT defined to establish the OEF context was

> *M autonomous UAVs cooperate to locate, identify, and track N stationary and moving ground targets and adapt to varying numbers of targets, the ratio of targets with known to unknown locations, to changing threats, to disrupted communications, and to changing "accuracy" requirements while maintaining continuous tracking.*

Within this context, researchers were to focus on three major problems crucial to the deployment of effective multi-agent systems.

1. **Coordination:** communication between the UAV-agents of local information and goals with the intention of improving the group's performance. The major MAS problem addressed in this topic is *design* interoperability, which is the necessity for the agent developer to use a common semantics and syntax for all agents that are to coordinate their actions. The UAV example is the development of techniques to assure that the UAVs share relevant information, such as individual and group situation assessment, individual goals, etc, that is needed to achieve group goals.
2. **Adaptation:** the ability to recognize and respond to unanticipated mission and environment dynamics by learning new behaviors. The MAS problem addressed is system *design* incompleteness, which is the recognition that the agent developer cannot and should not anticipate and design for all possible situations a set of agents might encounter. A UAV example is to respond to new threats, such as handheld surface-to-air weapons, to environment conditions, such as sand storms, to new missions, and to new objectives.
3. **Autonomy:** allow heterogeneous units to operate autonomously while fulfilling individual local goals and missions and yet cooperate effectively with the other agents to achieve the group goals. The MAS problem addressed is flexible, "run-time" *distribution of control*. The UAV example is to use new and existing UAVs to achieve critical missions through the distributed, designed regulation of collective behavior *without* introducing and using centralized command and control.

AIT intentionally specified the OEF abstract lyand to contain only minimal physical detail. AIT's purpose in providing such an abstract specification was to minimize details that might require significant effort but were irrelevant to the multi-agent system research topics of interest. Items such as path finding, terrain following, communications details, etc were such details. However, to give the OEF sufficient substance for research, AIT defined five problems of increasing complexity focused on Unmanned Aerial Vehicle (UAV) surveillance and reconnaissance tasks.

UAV-S (1) Baseline: Stationary Targets*:* coordinate the surveillance and reconnaissance of stationary targets in dynamic environments. The target locations may be known, which is coordination of the surveillance service, or unknown, which is coordination of the reconnaissance service. To create the dynamic environment, new stationary ground targets appear – popup – randomly and may require surveillance by a specified time. UAVs coordinate to minimize the time a popup target waits before they survey or find it and to minimize "wasted effort", that is, multiple surveillance or reconnaissance of the same target by different UAVs.

UAV-S (2) Cross-Mission Tasking: two or more UAVs must coordinate to service the same target nearly simultaneously with different sensors or from different angles. This problem, an extension of the baseline problem, is a harder MAS coordination problem since the UAVs must exchange and process information sufficient for them to arrive nearly simultaneously under conditions appropriate to maximize the usefulness of the data gathered and not merely to avoid ever approaching an already serviced target.

UAV-S (3) Imperfect Information: include several explicit sources of uncertainty, e.g. in the detection and identification of targets, as an extension to the baseline problem. The previous problems contained only implicit uncertainty, e.g. due to stale and/or partial information. Situations that one might investigate under this problem could include malfunctioning and mis-calibrated sensors and software problems. Additionally, communications between UAVs may be corrupted and incomplete, or enemies may inject false communications.

UAV-S (4) Mobile Targets: maintain identification, position, and speed estimates of mobile targets across the Area of Operations (AOR) in which the MAS of UAV-agents operate. This MAS coordination problem is both harder because it is a distributed multi-target data fusion-tracking problem and contains uncertainty that is more explicit. UAVs must pass processed and raw tracking information to each other and most likely must coordinate target handoffs to maintain maximum and efficient coverage of the entire AOR. One could naturally extend this problem to intelligent adversaries that attempt to avoid discovery and surveillance.

UAV-S (5) Hybrid Tasking: multi-mission, highly-domain-relevant problem combining some or all aspects of the previous problems. Such hybrid tasking problems can include intelligent stationary or mobile targets with the capability of destroying UAVs, to interrupt and disable UAV communications, cross-mission tasking of mobile targets, etc.

A primary object of the OEF process was to provide a means for potential users of the MAS design approaches developed by the TASK researchers to determine when one approach might be preferable to another. AIT facilitated many teleconferences with the TASK PI investigators and discussions at PI meetings about such research *metrics*. From these discussions, AIT developed a *parameterization* approach that can yield a quantitative basis for users to select an appropriate MAS approach for their particular requirements. The first element in AIT's parameterization is to decompose the TASK OEF context into four elements: (1) problem space parameters, (2) solution goals, (3) design space parameters, and (4) metrics.

Problem Space is *the quantitative specification of the problem to be solved*. A researcher specifies the particular problem in parameters derived from the following general *Problem Space* meta-parameters.

- **Scale** is the total number of items per unit of time with which the MAS must deal. Generally, *items* are the tasks the MAS must service such as the UAV surveillance and reconnaissance targets.
- **Communications** includes the quantifications of the type (such as line-of-sight, beacon, satellite, and laser) available bandwidth, ranges, and frequencies and the likely content accuracy and content completeness. The researcher must quantitatively specify the likely effects that items such as landscape features, anticipated atmospheric conditions, and possible jamming will have upon communications.
- **Information Density** is the amount of information and data available to the system both statically and dynamically. Static information includes terrain maps, sensor capabilities, and the capabilities of other agents. Dynamic data includes the sensor data, communications content, and for UAVs flight data.
- **Resources** include the amount of computing each agent and the MAS can perform. For example, explicitly included are parametric quantifications of storage, computing throughput, and the type and amount of "sensing" each agent can perform
- **Uncertainty** is a quantification of the incomplete, ambiguous, and erroneous information that exists within the Problem Space.
- **Threat Level** is a quantification of the number and type of entities opposing the MAS.
- **Dynamics** are the rates at which the Problem Space Parameters change and the rates of change of these rates of change. For example, the anticipated rate at which popup targets appear is one of these dynamic parameters as is the rate at which this rate changes.

Solution Goals are *the specifications for a "successful" solution predefined in terms of measurable parameters*. A researcher must derive the measurable parameters from meta-parameters that generally characterize *Solution Goals*.

- **Purpose**: quantification of how well the MAS performs the tasks for which it was designed. For example, in the UAV surveillance and reconnaissance problems, one might have designed the MAS to survey 10 targets per minute at maximum load with a tolerance of ±0.5. If the MAS survey a maximum of 8.76 targets per minute, it is not fulfilling its purpose with respect to the *Solution Goals*. One assumes that the basic goals of the MAS are fulfilled and do not require quantification. For example, a MAS that is to perform surveillance of stationary ground targets actually surveys stationary ground targets.
- **Accuracy**: specified quantification of the tolerances permitted in the individual datum and information that the MAS actually delivers. For example, the goal for a MAS may be to determine the tracks of all moving targets to less than 1 meter.
- **Timeliness**: the MAS delivers information and data and performs activities and tasks at specified time goals and within specified time windows.
- **Survivability**: after catastrophe, the MAS continues to meet the purpose, accuracy, and timeliness goals within pre-specified limits.
- **Robustness**: the MAS continues to meet the purpose, accuracy, and timeliness goals within specific limits when resources degrade, problem complexity changes, and problem ambiguity or environmental uncertainty change.

- **Stability**: the MAS continues to meet purpose, accuracy, and timeliness goals within specified limits as the *Problem Space* parameters fluctuate widely and rapidly.

A *Design* is *a set of related algorithms that purports to meet the solution goals defined by the Problem Space parameters*. A researcher must specify his design or class of designs in terms of parameters derived from the **Design Space** meta-parameters.

- **MAS Size**: the number of agents that make up the MAS.
- **Determinism**: a measure to which the MAS is *deterministic* or *non-deterministic*. Note: The MAS might be non-deterministic even if the individual agent behaviors are deterministic, such as is true with swarms. Similarly, lowest-level behaviors of a single agent might be deterministic or reactive or, but if the agent processes data and information based on its recent history rather than just its current situation, the overall behavior of the agent and MAS may be non-deterministic.
- **Complexity**: simple static complexity, such as *program size*; simple deterministic dynamic complexity, such as number of elementary steps before halt (time) or total amount of "memory" used before halt (space – cell used in a Turing machine, e.g.); simple non-deterministic dynamic complexity, such as the number of steps in the *shortest* acceptable computational path before halt (time); similarly for "memory" (space).
- **Completeness**: which solution goals does the design met under all appropriate and allowed combinations of problem space parameters
- **Degree of Learning**: applies to both individual agents and to the system of agents
- **Connectedness**: the degree to which an agent's beliefs and actions are influenced directly by the *internal* state of is neighbors, other agents, and the multi-agent system as a whole, and the level of detail to which the agent examines this external information.
- **Knowledge**: the extent to which knowledge, beyond the implicit knowledge captured within the structure of the agents and of the multi-agent system, is used by the agents in forming their beliefs and/or action, and the nature of this knowledge: global and static, dynamic and local, and/or combinations.

The **Metrics** are the *bases for measures of how well the solution goals are satisfied under the various allowable combinations of values from the problem space parameters' domains*. *Metrics* of this time are what one measures during experiments, quantized by the *Problem Space* parameters, for particular MAS implementations, quantized by the *Design Space* parameters.

- Real-time and summary measurements of the Problem Solution Goals: *Purpose, Accuracy, Timeliness, Survivability, Robustness, Stability*
- Plus additional problem specific goals
- Plus additional design-implementation specific goals
- The *Autonomous Control Level* (ACL) Metrics
  - Developed by Bruce T. Clough, Technical Area leader: Control Automation, AFRL/WP and colleagues
  - *Metrics, Schemetrics! How do you track UAV's autonomy?*, AIAA-2002-3499.
  - Incorporated into the DoD's *Unmanned Aerial Vehicle Roadmap 2000-2025*, Office of the Secretary of Defense, Washington DC. April 2001
  - In some sense the ACL Metrics are an instantiation of the abstract Design Space meta-parameters for the real, operational UAV MAS problem

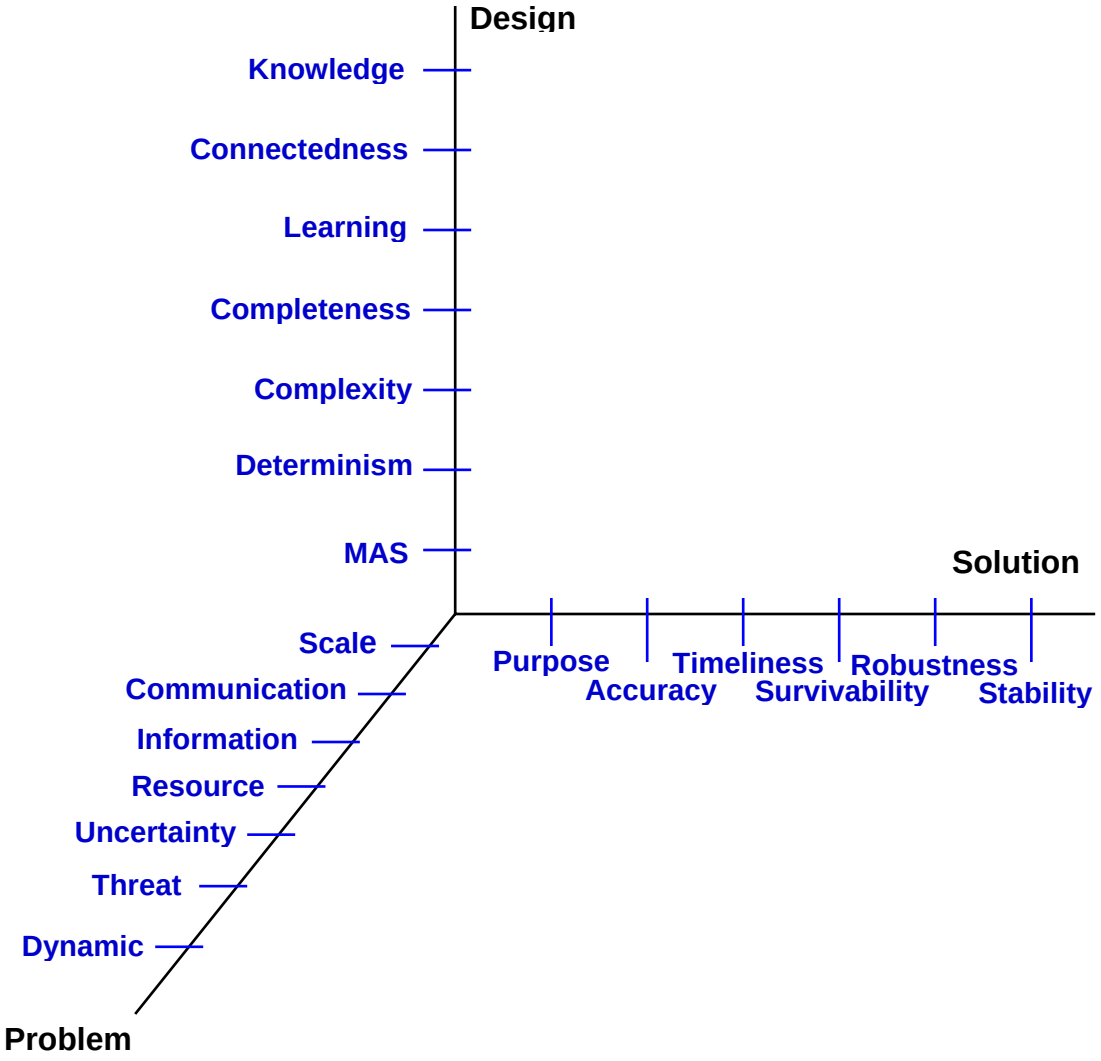

Figure 1: The Problem, Solution, and Design Spaces of Multi-Agent Systems

These parameterizations of the problem space, solution goals, and design space are consistent with the principal multi-agent system problems of coordination, adaptation, and autonomy and with the metrics that measure the results of a multi-agent system implementation. Future research into these parameterizations and metrics should focus on specific problems, such as the OEF surveillance and reconnaissance problems, and specific quantification of the associated parameters. Treating explicit quantitative parameterizations of the problem, solution, and design spaces as independent variables of the dependent variables of the performance metrics and using experimental simulations to determine the functional dependencies is a seldom-realized goal of systems engineering. These investigations for the OEF, conducted as discussions between the TASK Principal Investigators and facilitated by AIT, are a unique contribution. Further development and exploration in other DARPA programs can provide a formal basis for this primary systems engineering activity.

3 MULTI-AGENT SYSTEM (MAS) RESEARCH

AIT's MAS researched focused on two areas. The first focused on a distributed, dynamic algorithm that can use collaboration messages between UAV-agents to perform the surveillance tasks described in the OEF problems **UAV-S (1)** and **UAV-S(2)**, that is, popup stationary ground site targets and cross-mission tasking. Section 2.1 contains description and discussion of this work. The second focused on some preliminary investigations into the use of Bayesian networks as an approach to implementing plans in MAS teams that accommodate adaptation to incomplete and unknown knowledge.

3.1 Starvation and Waiting Times for a Distributed and Dynamic Vehicle Routing Problem

3.1.1 Introduction

Recent technological advances in communications and the advent of unmanned aerial vehicles (UAVs) with the ability to self-navigate has resulted in challenging new problems becoming relevant to researchers in the operations research (OR) community. It is now possible (or shortly will be possible) for a group of UAVs to be implementing a plan in an area that is remote compared to the location of other entities of the system, such as a base station or dispatcher. In such a case as this, it may be infeasible or too costly for the dispatcher to maintain a complete situational awareness on the status of the UAVs and the overall plan. If the dispatcher has limited information about the UAVs' operating environment, then it is best that the UAVs autonomously decide what actions to take in order to satisfy the overall plan. This problem is further complicated when it is assumed that each individual UAV has limited knowledge about the environment. Due to restrictions on the size and cost of a UAV, each UAV may have limited ability to observe the environment and communicate with other UAVs. This paper investigates a problem that contains all of these aspects.

The motivation for this paper comes from a military application. Consider multiple heterogeneous UAVs, operating in an area of operations (AO) Ω of fixed specifications, that need to perform surveillance of objects (tasks) that are located at fixed positions (i.e., the tasks are stationary). The UAVs have different sensors to perform surveillance of tasks of different types, but, generally, not all UAVs have all of the different types of sensors. Each task is one of a finite number of types and it must be serviced by a sensor of the same type. The AO and surrounding region is hostile; making it impossible to locate a dispatcher or central command nearby. The dispatcher becomes aware of tasks that need to be surveilled through sources of intelligence and then relays the information (location of task and task type) to all of the UAVs. Therefore, the UAVs become aware of the tasks in a dynamic manner and, in modeling this problem, it is assumed that the tasks arrive one at a time, where the task inter-arrival times are independent and identically distributed. Also, when this problem is modeled, it is assumed that the locations of the tasks are independent and identically distributed. The UAVs are compact with limited ability to communicate and thus cannot send information back to the dispatcher to indicate their status, such as their current locations and lists of serviced tasks; thereby implying that the dispatcher is ill-suited to give commands to the UAVs. Furthermore, the UAVs can only communicate with other UAVs within a specified range (communication range) and can only

observe the locations of other UAVs within a specified range (observance range). Therefore, each UAV must individually decide for itself which tasks to service.

For many particular scenarios of this military application, the intuitive solution would be to take a "partitioning approach" and assign each vehicle a distinct area to "cover." As an example, suppose that there is only one task type, task locations are uniformly distributed throughout the AO, and each vehicle can service any of the tasks. In this case, a good approach to the problem is obtained by simply assigning each UAV to a distinct region, where the regions are of approximately equal area, and force each UAV to only service tasks in its region. Bertsimas and van Ryzin [1] conjecture that such a policy is optimal when an optimal single-vehicle policy is used in each region.

The focus of this research is to develop algorithms for scenarios that *cannot* be addressed with a partitioning approach as described in the previous paragraph. This paper introduces a control policy to address the military application described earlier for cases where there are multiple task types and heterogeneous UAVs. To investigate the effectiveness of the control policy, it is tested on a scenario that is illustrative of scenarios requiring interaction among the UAVs. This scenario consists of five task types (denoted y_1, y_2,…, y_5) and four UAVs, where tasks of type y_1 can be serviced by all of the UAVs and each of the remaining four task types (y_2, y_3, y_4, y_5) have exactly one UAV that can service them. Tasks of each type are distributed uniformly throughout the AO. Since there is only one UAV for each of the task types y_2, y_3, y_4, or y_5, a partitioning approach is infeasible for this problem (i.e., each vehicle needs to be able to fly everywhere in the AO).

The objective for this military application is to service the tasks in a manner that is as efficient as possible over a given time horizon, where efficiency is specified by a function of the task waiting time distribution. The waiting time of a task is the time between when the UAVs become aware of the task and the time of service for the task. The task waiting time distribution is the probability distribution of the waiting time for a randomly selected task. A natural measure of efficiency is given by the expected waiting time for a randomly selected task. The expected waiting time for a randomly selected task that results from a given control algorithm can be estimated by simply computing the average waiting time for the tasks in a simulated environment (for some control policies, the expected waiting time can be computed analytically). If the tasks are serviced so that the expected task waiting time is minimized, then it is possible that many tasks will wait an exceedingly long time for service. This problem arises since a newly arriving task may be more favorable to service than a task that has waited for a long time and is far away from the current locations of the UAVs. Therefore, a more reasonable objective for this problem may consist of jointly trying to minimize the expected waiting time and the probability that a task waits more than T_w seconds for service, where T_w is a specified constant. This paper introduces an algorithm according to this objective. A formal definition of a system performance measure is given in Section 2.

In this paper, a model is proposed that captures the important aspects of the military application and then a purely decentralized control approach is introduced to address this model. An approach to the military application can be broadly categorized as being either a decentralized or centralized technique. A centralized technique usually contains a central node that gathers as

much information as possible (according to time constraints and costs) about the environment and then issues commands to all of the UAVs based on this information. A technique that consists of periodic "meeting" times or other methods to consolidate information at a central location for the purposes of decision making is a centralized technique. The term "meeting" is meant to refer to the situation where multiple UAVs fly to a designated region in order to share information and possibly coordinate future actions. While the centralized approach may be successful for many scenarios, this paper is focusing on scenarios that are better addressed in a purely distributed/decentralized fashion. Situations involving rapidly changing environments (high arrival rate of incoming tasks) are probably addressed more efficiently with purely decentralized approaches. A purely decentralized technique implies a control policy for an individual UAV that *does not* consist of periodic "meeting" times between the UAVs. Note that the regularly scheduled meetings of a centralized technique incur a cost; it takes time and resources for the UAVs to travel to the designated "meeting" area. This paper does not undertake a systematic procedure to determine under what scenarios the decentralized techniques outperform centralized techniques; that is left for future research. However, this paper does compare several decentralized approaches in terms of their resultant task waiting time distributions.

The dynamic military problem of this paper is addressed by solving a sequence of static optimization problems. In our approach, each UAV re-optimizes its schedule after every T_s seconds has passed (T_s is specified and fixed) and whenever it becomes aware of a new task. After each time step (of length T_s), the UAVs select tasks that they plan to service in the near future. The intent is for the UAVs to select tasks that are unlikely to be selected by other UAVs and would result in desirable operating characteristics for the overall system. Once the tasks are selected, each UAV solves a static optimization problem to determine the order in which the selected tasks should be serviced. The objective function for the optimization problem is defined according to the desired characteristics of the overall system. If the system-wide goal is to minimize the average task waiting time, then the objective for the static optimization (minimization) problem could be to minimize the average waiting time of the selected tasks. As another example, to prevent tasks from starving (waiting an exceedingly long time for service), the objective function of the static optimization problem could place a penalty on un-serviced tasks that have waited more than a specified amount of time.

The static optimization problem solved at each time step and after each task arrives is a new (to the best of our knowledge) combinatorial optimization problem and will be referred to as the generalized minimum latency problem (GMLP). GMLP contains the minimum latency problem (also called the traveling repairman problem) as a sub-problem. GMLP is defined by a graph, a specified starting node, and a penalty function $P_v(t_v)$ ($t_v \geq 0$) associated with each vertex v. The objective of GMLP is to find a path that visits each vertex at least once in such a way that the sum of the penalties is minimized. An area of future research includes a theoretical investigation of the complexity of this new combinatorial problem. The penalty for visiting a vertex v at time t_v is given by $P_v(t_v)$. GMLP is formally defined in Section 2.

Due to the dynamic nature of the problem, there can be many cases where the UAVs need to make a decision quickly. In an environment where tasks are arriving at a very high rate, the UAVs must quickly decide whether or not an incoming task should be serviced immediately, and

the decision of whether or not to service a task would depend on other tasks already in the system. We have made a conscious effort to use algorithms for the static optimization problem that can operate quickly. At this stage of our research, it is hard to precisely specify what is quick or efficient. However, we are able to show that our algorithms provide good results even though the optimization is limited to a few iterations of a local search algorithm. We believe that the algorithms given could be incorporated to work in real-time.

In Section 2, the military application that was given earlier is defined as a combinatorial optimization problem called the multiple autonomous vehicle, dynamic traveling repairman problem (MAV-DTRP). MAV-DTRP is a generalization of the multiple vehicle-DTRP (MV-DTRP) proposed by Bertsimas and Van Ryzin [1,15]. The MV-DTRP is a particular type of stochastic and dynamic vehicle routing problem (see [20] for a recent survey on deterministic vehicle routing problems as well as dynamic and stochastic variants) that consists of demands (tasks) arriving dynamically over time, where these demands require an independent and identically distributed amount of on-site service by a vehicle. The objective is to find a control policy for the vehicles that minimizes the average waiting time for demands to be serviced, over an infinite horizon. Bertsimas and Van Ryzin [1,15], analyze several policies for MV-DTRP and analytically compute their average waiting times. For some other policies, they obtain bounds on the average waiting time or estimate the average waiting time through simulated experiments. They consider two different situations, vehicles with unlimited capacity and vehicles with limited capacity. If a vehicle has a capacity of q, then the vehicle can serve at most q demands before returning to a depot. In an earlier paper [2], Bertsimas and Van Ryzin introduce and analyze the DTRP, which is the MV-DTRP consisting of a single vehicle.

MAV-DTRP generalizes MV-DTRP by considering a measure of performance that is a function of the task waiting time distribution rather than just focusing on the expected waiting time. Also, MAV-DTRP restricts the vehicles to make autonomous decisions in the face of uncertainty. That is, the vehicles in MAV-DTRP model have limited ability to detect the location of other UAVs and limited ability to communicate. The MV-DTRP model does not consider situations involving uncertainty in terms of vehicle locations, future decisions of vehicles, and previously serviced demands by vehicles.

MAV-DTRP is a particular type of Dynamic Vehicle Routing Problem (DVRP) (see [7], [8], [12] and [16] for an overview). Formally, MAV-DTRP is a dynamic problem, but it is not considered to be a stochastic vehicle routing problem (SVRP) [17]. Psarftis [16] proposes a taxonomy for characterizing attributes of information in vehicle routing problems. The information available in a vehicle routing problem can be characterized by the evolution of information (static vs. dynamic), quality of information (known vs. probabilistic vs. unknown), availability of information (local vs. global), and processing of information (centralized vs. decentralized). Static inputs refer to information that is known throughout the duration of the routing process, whereas dynamic inputs are revealed to the system over time. The quality of information for an input corresponds to the amount of uncertainty about the input. As an example, the on-site service time of a customer (demand) in a routing problem may be known exactly by the vehicles or it may be known to follow a probability distribution or it may be completely unknown. If an input to a routing problem is known probabilistically, then the problem is called a SVRP. The information can also be local or global. If an input contains

global information, then the information is known to the entities of the system at all locations. Local information corresponds to inputs that are not available globally to all entities of the system. The information given by the inputs can be processed in a centralized fashion, in that all information is sent to a central unit so that a decision can be made with all available inputs. On the other hand, information can be processed in a decentralized way if the vehicles are allowed to autonomously determine which actions to take. A decentralized approach may be indicated in a system where information is local and there is a significant cost (monetary cost or in terms of time) of processing information at a central unit. The MAV-DTRP, which is the object of this paper, is a dynamic problem, where some information is considered unknown (such as vehicle locations and whether or not a task has been serviced), some information is known locally (each vehicle contains information that may be unknown to other vehicles and the other vehicles can only obtain the information by moving to within communication range), and information is processed locally (the vehicles are autonomous). MV-DTRP is a stochastic and dynamic problem, where information is known globally and can be processed centrally without any consideration for communication costs.

Dynamic and stochastic routing problems are starting to receive more attention from researchers since they are realistic representations of many applications. For example, the travel time between two locations in a major metropolitan area can be modeled by a random variable (some days the traffic is worse than others). Also, in many situations, such as those that occur in dial-a-ride problems, not all information is known before the routing process begins so that some information is revealed over time (new requests for rides are received). DVRPs are considered to be a subclass of a larger class of problems, called dynamic transportation problems [8]. Several examples of dynamic transportation problems follow. An instance of the dynamic traveling salesman problem (DTSP – see [23] and [24]) consists of a group of locations, where customers arrive at these locations and wait for service, and the goal is to find a policy that minimizes the average waiting time for the customers. In DTSP, the inter-arrival times of customers are assumed to be independent and identically distributed and an arriving customer is placed at location i with probability p_i. Powell [21] and Powell et al. [9] introduce and develop algorithms for the stochastic dynamic assignment problem. Also, Powell et al [3] discuss a dynamic truck load problem with user noncompliance. User noncompliance refers to the system not following the decisions purported by the optimization routine. Situations such as these could occur due to the stochastic and random nature of the dynamic truck load problem. For instance, the problem they study assumes that the travel times between points are random variables and therefore part of a solution may become infeasible due to a late arrival of a vehicle. They indicate that the user noncompliance in real dynamic truck load systems may be as high as 30%, which clouds the picture of defining "optimality." In other words, how can a solution be optimal if 30% of it will not be followed? Powell et al. address the problem by solving a sequence of static optimization problems. They show that in a random environment, in some situations, it is better to find near-optimal solutions to the static optimization problem rather than finding a globally optimal solution. Minkoff [19] introduces the delivery dispatching problem (DDP), which is a stochastic and dynamic problem consisting of a set of fixed customers at specified locations that maintain an inventory of a particular good. The customers lose inventory during every time stage according to some probability distribution. The goal is to replenish the customers' demands so as to minimize the long-run average cost (transportation costs plus inventory costs plus costs for unmet demand). The customers receive goods from a fleet of

vehicles that each travel according to some itinerary (route through a subset of the customers together with the amount of goods to deliver to each customer on the route). At each time step, a decision is made as to the number of vehicles to send out and which itineraries to assign to the vehicles. Minkoff models this as a Markov decision process, where the system states consist of the inventory level at each of the customers and the objective is to determine which itineraries to use for each possible state of the system. Moizumi and Cybenko [11] define the Traveling Agent Problem (TAP) and introduce efficient algorithms for the problem. TAP is a stochastic problem that consists of a set of locations and a probability for each location that represents the likelihood of "success" when the agent visits that location. The objective of the traveling agent problem is to find a path through the locations that minimizes the time it takes to have a "success" or else determine that all locations are "failures." As an illustrative example, they mentioned a person that needed to buy a particular item and had several stores to choose from, where each store had a certain probability of containing the item, and the person wanted to minimize his/her total searching time for the item. All of these papers have some commonality with the work presented here, though (to the best of our knowledge) there are no papers that address a variant of DVRPs and consider situations that involve uncertainty in vehicle locations, vehicle plans, or the tasks previously serviced by the vehicles.

There are quite a large number of papers that propose approaches to particular dynamic or stochastic transportation problems. In this paragraph, a few of these papers are discussed. Papastavrou [18] studies DTRPs and proposes a policy that performs well in both light and heavy traffic. Papastavrou shows that his policy is asymptotically optimal in light traffic and within a constant factor of the optimal policy for heavy traffic. He models the state of the system as a branching process with state dependent immigration. Also, Ichoua [25] et al. propose a solution to a dynamic and vehicle routing problem that probabilistically considers the effect that decisions have on the future state of the system. Gendreau et al. [22] propose an adaptive neighborhood search (tabu search) algorithm to address a dynamic vehicle routing problem with pick-ups and deliveries. They discuss how their approach can be implemented in a parallel fashion in order to reduce the execution time of the algorithm. Finding quality solutions in a quick amount of time is very important for dynamic problems, since decisions have to be made rapidly when new information is made available to the system.

The paper is organized as follows: Section 2 contains the formal definitions of MAV-DTRP and GMLP, and the formulation of our approach to MAV-DTRP. In Section 3, the simulation environment is discussed in detail and the experimental results are given. The material in Section 4 analytically computes the average waiting time for some control policies of MAV-DTRP. We compare these results to the experimental results of Section 3. Finally, Section 5 contains a discussion of future work and provides a conclusion.

3.1.2 Problem Formulation and Approach

First, MAV-DTRP will be formally defined. Later in this section, our algorithmic approach to this problem will be specified. The military application that motivated the definition of MAV-DTRP consists of UAVs that operate in a three-dimensional world. However, to simplify the problem, the vehicles here are assumed to operate in a plane. Furthermore, it is assumed that the vehicles cannot crash into each other (i.e., two vehicles are allowed to occupy the same position at the same time).

MAV-DTRP: An instance of this problem is described by the following: A group of m vehicles are needed to service tasks (demands) that arrive dynamically over time. Each task generated is of a particular type (task types are denoted by $\{y_1,y_2,\ldots,y_k\}$) and tasks are generated up to a time T. Tasks of type y_i (for $i = 1,2,\ldots,k$) arrive according to a Poisson Process with rate λ_i. All tasks are located in a two-dimensional, convex region Ω. When a task of type i is generated, it is placed in Ω according to a spatial distribution $g_i : \Omega \to R^+$. Every UAV has a collection of sensors, which are used to service the tasks. Let $\{s_1,s_2,\ldots,s_k\}$ denote the k different types of sensors, where a task of type y_i must be serviced by a sensor of type s_i. Each vehicle has a specified range for communicating with other vehicles (communication range), observing the location of other vehicles (observance range), and servicing tasks (sensor range). The sensor range specifies how close a vehicle needs to be to a task in order to service that task. The goal is to service all tasks so that a prescribed objective function is minimized. Let W_u be the random variable equal to the waiting time for a randomly selected task, under the policy u, and f_u be the probability density function for W_u. The objective function $F:\Theta\to R$ (where Θ is the set of all probability density functions defined over $[0,\infty)$) is defined to be a function of the probability density f_u and the goal is to find the policy u that minimizes $F(f_u)$.

It is not the purpose of this paper to find policies that minimize particular objective functions F. Rather, this paper shows how different policies can be used to alter the resultant characteristics of the density function f_u. The most natural example for the objective function would be to define it to equal the expected waiting time for a randomly selected task. In this case, the function F would be defined by

$$F(f_u) = \int_0^\infty t f_u(t)\,dt = E(W_u).$$

In this case, note that there is a unit penalty applied to the system for each second of the expected waiting time. As another example, F could be defined to be

$$F(f_u) = \int_0^\infty t f_u(t)\,dt + M\int_0^\infty (t-w) f_u(t)\,dt = E(W_u) + ME(\max\{0, W_u - T_w\}),$$

where w and M are specified constants. The purpose of this objective function is to penalize algorithms that allow a large number of tasks to wait for more than w seconds before being serviced. This objective function specifies a unit penalty to the system for each second of the expected waiting time and M units of penalty for each second of the expected time the tasks wait more than the threshold T_w.

Note that MAV-DTRP could be defined over an infinite horizon. In that case, the objective becomes to minimize a function of the task waiting time distribution in the long-run. To define a problem in this way so that it makes sense, attention must be restricted to algorithms that guarantee, with probability one, that all generated tasks are serviced. Otherwise, it would be possible to define an algorithm that only services tasks that are favorable, according to the objective function, and ignore other tasks. For instance, if the objective is to minimize the average waiting time, then an algorithm can be defined that only services tasks that are generated with a distance ε of a specified location. By reducing the value of ε, an algorithm can be easily obtained with arbitrary small expected task waiting time.

The CTS approach is proposed to address this problem. With this approach, MAV-DTRP is handled by solving a sequence of static optimization problems. The pseudo-code for CTS follows. Note that CTS is a policy for a single vehicle operating in an environment consisting of other heterogeneous vehicles. Therefore, by having each vehicle make decisions from the CTS policy, we have a decentralized approach to MAV-DTRP.

CTS

time = 0;
L = [ ];
nextUpdate = 0;
timeBetweenUpdates = T_s;
For each simulation step
    Let Δ equal all UAVs within the observance range of this vehicle
    If (*time* >= *nextUpdate*)
        Let *L* equal all tasks that are closer to this vehicle than any other vehicle in Δ
        Let *solution* equal a permutation of *L* that represents the order the tasks in *L* should be serviced. This is obtained by an optimization routine **A**(*L*)
        nextUpdate = *nextUpdate* + *timeBetweenUpdates*;
    else
        If a new task has arrived and this vehicle is closer to it than any other vehicle in Δ, then add it to list *L* AND let *solution* be the permutation obtained from **A**(*L*)
    end
    time = *time*+ *timeStep*; // *timeStep* is amount of real-time associated with a simulation step
    if (length of *solution* > 0)
            move to the location of the next task, based on *solution*
    else
        follow Greedy policy // Greedy policy is discussed in Section 3
    end
    Remove any serviced tasks from *L* and *solution*
    Communicate new information to other vehicles in communication range
end of for loop

The CTS approach as described above guarantees that each task is selected by a vehicle (possibly multiple vehicles) at the beginning of every time step. This is an important characteristic, since otherwise we would have to worry about tasks being repeatedly left un-selected by all of the vehicles. It is possible to develop an approach that allows a vehicle to probabilistically estimate the location of other vehicles outside of its observance range. Also, it is possible to develop a technique to estimate the likelihood that a task has been serviced by another vehicle. However, developing techniques for these two estimation problems is nontrivial. For example, it is doubtful that there exists an effective approach for estimating locations of vehicles that have not been in the observance range for a long time. Furthermore, it is not obvious that estimation techniques for these two problems would improve performance since estimation error could have a seriously negative impact on the problem.

The optimization routine **A**(.) is now described. This optimization routine determines the schedule or order of service of the tasks in L. The optimization algorithm **A**(.) finds a near-optimal solution to instances of GMLP. An instance to GMLP is represented by a graph $G = (V,E)$, specified starting vertex $s \in V$, set of travel times t_{ij} between all vertices i and j, and a penalty function $P_v : R \rightarrow R$ associated with each vertex $v \in V$. The objective is to find a path for a vehicle starting at s, which visits each vertex at least once and minimizes $\sum_{v \in V} P_v(t_v)$, where t_v is the time the vehicle reaches vertex v. The number $P_v(t)$ represents the amount of information (or value or money) lost for visiting vertex v at time t. In this paper, GMLP is restricted to cases where the penalty function increases with time, which denotes the fact that it is better to service a task earlier as compared to a later time. It is easy to see that if the penalty function satisfies, for every instance, $P_v(t) \equiv P(t) = at$ (for some number $a > 0$), then the resulting problem is equivalent to the MLP. MLP can be approximated in polynomial time [4], but it is MAX-SNP hard, which implies that it is unlikely for there to exist a polynomial-time approximation scheme (PTAS) [6] for this problem (unless P=NP). Note that since MLP is a sub-problem of GMLP, then GMLP is also NP-hard [5]. The following lemma shows that restricted versions of GMLP are in APX [6] (i.e., it can be approximated in polynomial time within a constant factor $\alpha \geq 1$).

Lemma 1: Define GMLP(a, b) (where $a < b$) to be the set of instances of GMLP where $at < P_v(t) < bt$ for all $t > 0$ and vertices v. Then GMLP(a, b) can be approximated within a factor of $(b/a)\alpha$ in polynomial time, for some constant $\alpha \geq 1$.

Proof: Let $G = (V,E)$, starting vertex $s \in V$, set of travel times t_{ij} between all vertices i and j, and penalty functions $P_v : R \rightarrow R$ for all $v \in V$, define an instance of GMLP(a, b). Let $\omega = sv_{11}v_{12}\ldots v_{1n}$ represent an optimal solution to this instance. Then, there exists a constant α so that a α-optimal solution $sv_{21}v_{22}\ldots v_{2n}$ can be found (in polynomial time) to the corresponding MLP. It follows that

$$\frac{\sum_i P_{v_{2i}}(t_{v_{2i}})}{\sum_i P_{v_{1i}}(t_{v_{1i}})} \leq \frac{b\sum_i t_{v_{2i}}}{a\sum_i t_{v_{1i}}} \leq \frac{b}{a}\alpha .$$

Since the solution $sv_{21}v_{22}\ldots v_{2n}$ can be found in polynomial time the result follows.

The GMLP model is used by the optimization routine **A**(.) in the following way. A penalty function is associated with each *task* that represents the amount of information (or value or money) lost as a function of time. All tasks are considered to have the same priority so that we let $P(t)$ represent the penalty function for all tasks. The function $P(t)$ is When the algorithm **A**(.) is called, an instance of GMLP is constructed in the following way: a graph $G = (V,E)$ is constructed where each vertex corresponds to either the location of a task or to the current location of the vehicle. The starting vertex *s* corresponds to the vehicle's current location. The travel times between all pairs of tasks and between the starting location and all of the tasks is estimated to be the distance between the two locations divided by the cruising speed of the vehicle. The penalty function of vertex $v \in V$ is given by $P_v(t) = P(t + t_{ov})$, where t_{ov} is the time that task *v* was generated. The algorithm **A**(.) then finds a solution with near-minimal value for $\sum_{v \in V} P_v(t_v)$, where t_v is the additional amount of time that vertex (demand) *v* will have to wait, from the *current time*, for service.

Due to the dynamic nature of MAV-DVRP, it is important to develop algorithms that run in near real-time. The importance of very efficient algorithms is magnified by the fact that, in reality, the UAVs or vehicles will be compact and possess very little computational power. In other words, it is desirable to have an algorithm that runs extremely fast and is yet still able to produce quality solutions. Therefore, a local search algorithm is proposed to address GMLP. For many combinatorial optimization problems, the performance of local search algorithms is greatly improved by investing some time to find a high-quality initial solution. Since the number of computations is severely limited for these experimentations, the importance of finding a high quality initial solution is intensified.

In this paper, the 2.5-opt local search algorithm [10], which is a well-known technique for MLP and TSP, is used to address GMLP. In general, the *k*-opt neighborhood of a Hamiltonian path ω is defined to be the set of all Hamiltonian paths that can be obtained by exchanging at most *k* edges. The 2.5-opt neighborhood consists of the 2-opt neighborhood together with some of the three edge exchanges. In the experiments given in Section 3, the 2.5-opt neighborhood was used to find a near-optimal solution of the static optimization problem. The 2.5-opt local search algorithm was applied to 6-10 starting solutions. About half of the starting solutions were chosen at random and the other half was a high-quality initial solution chosen by a greedy heuristic. The implemented local search algorithm examines all solutions in a neighborhood and moves to the best neighbor found, if the current solution is not a local optimum. The algorithm stops once a local minimum is found, and then restarts with another initial solution. To prevent the algorithm from running to long, the number of iterations of the local search algorithm was limited to between 10 and 20, for each initial solution. The number of iterations of the algorithm is restricted so that solutions can be found quickly (near real-time) and due to the fact that the static optimization problem usually involved a small number of tasks.

3.1.3 Distributed Task Scheduling with Real-Time Optimization

This section provides experimental results for a particular instance of MAV-DTRP. Experimental results are given for the CTS and Greedy approaches over the particular instance. These results demonstrate that the CTS approach is superior to the Greedy approach for the given instance and indicate that it is a viable approach to MAV-DTRP. The results are given for different arrival rates and observance ranges.

The experimental results given later in this section were obtained from ALPHATECH Inc.'s Testbed for Taskable Agent Systems (TTAS). TTAS is a simulation testbed that provides a framework for examining autonomous, cooperative, and task-able agents within a simulated three-dimensional physical environment. TTAS provides a standard physically mobile container, which represents an unmanned aerial, ground, water surface, or underwater vehicle, that houses a *Vehicle Agent* implementation. Through application programming interfaces (APIs), which are standard for TTAS, the encapsulated Vehicle Agent controls the vehicle's movement about the three dimensional environment and its performance of tasks, such as an UAV's surveillance and reconnaissance tasks. Standard modules within a TTAS vehicle handle messages, autonomic functions (such things as navigation to a waypoint, obstacle and other vehicle collision avoidance), task service management, and plan management. A simple kinematics model encapsulates the dynamics of each type of vehicle (air, ground, water surface, underwater). If an experimenter requires higher fidelity dynamics, he can substitute a more complete physics engine; however, simple kinetics models are generally sufficient for experimental examination of the performance characteristics of the system. Similarly, TTAS provides simple models for sensors, such as various cameras and radars: again, an experimenter can substitute higher fidelity implementations if necessary. Simple tasks, such as static ground sites that only have locations, are provided, but since *tasks* are also agents, an experimenter can implement more general tasks, such as moving and evasive tasks that represent groups of troops. Generally, to carry out simulated experiments of a multi-agent system with TTAS, an experimenter need only design and implement his agents in conformance with TTAS's APIs and guidelines (TTAS supplies everything else). MAV-DTRP has been implemented into the TTAS framework in order to compare and evaluate different control policies.

Due to an intensive amount of computational time needed for the experimental runs, the results of this section are given for a *single* scenario that is illustrative of many situations. All of the experiments given in this section were run for 16 hours of simulated time, which translated into several hours of computational time on a 3.06 GHz Pentium 4. Every step of the simulation corresponds to 0.1 seconds of real-time. The scenario consists of four UAVs servicing five different types of tasks (task types denoted by $\{y_1, y_2, y_3, y_4, y_5\}$) in an AO of size 100km by 100km. Every UAV can service a task of type y_1, and there is only one UAV that can service any of the remaining types of tasks. Therefore, one UAV has sensors $\{s_1, s_2\}$, a second UAV has sensors $\{s_1, s_3\}$, a third UAV has sensors $\{s_1, s_4\}$, and the final UAV has sensors $\{s_1, s_5\}$. A generated task is of type y_1 with probability 0.8, and the other task types each occur with probability 0.05. Thus, if λ is the arrival rate for all of the tasks, then $\lambda_1 = 0.8\lambda$ and $\lambda_i = 0.05\lambda$ for all $i = 2,3,4,5$. The overall arrival rate was varied between experiments among $\lambda = 1/25$s and 1/15s. The observance range was also varied between experiments among 40km and 80km. All of the UAVs were defined to have a communication range of 20km and all sensors were accurate

within 1km of the location of a task. Therefore, a task of type y_i is considered to be serviced if a UAV, with sensor s_i, travels to within 1km of the task.

The purpose of these experiments is to compare the results obtained with the CTS approach against several baseline algorithmic approaches. The CTS approach is tested with different values of T_s and static objective functions. For the experiments of this section, the general form of the penalty function for task servicing is the following:

$$P(t) = t + M\max\{0, t - T_w\}.$$

Recall that the static objective function is defined to be the sum of the penalties for servicing all of the selected tasks. Note the correspondence between the static objective function obtained from using the above penalty functions and the MAV-DTRP objective function given by

$$F(f_u) = \int_0^\infty t f_u(t)dt + M\int_{T_w}^\infty (t-w) f_u(t)dt = E(W_u) + ME(\max\{0, W_u - T_w\}).$$

In some cases, the CTS approach is tested with a linear penalty function for task servicing (i.e., $M = 0$), which results in the static optimization problem being equivalent to the MLP. In this case, the purpose of the static optimization problem is to minimize the average waiting time (or total waiting time) for the tasks. The CTS approach is also tested with a piecewise linear penalty function for task servicing, so that a penalty is associated with tasks that wait more than a prescribed threshold for service (i.e., $M > 0$). This penalty function associates a "starving" penalty to tasks that have waited more than T_w seconds for service. The term M is called the starving penalty factor and the term T_w is referred to as the starving threshold.

Experiments were conducted for a baseline approach that is called the greedy approach. In the greedy approach, a UAV moves to the closest task it believes to be un-serviced that is not closer to another UAV. If all tasks UAV i believes to be un-serviced are closer to another UAV, UAV i will move to the closest task j that it believes is un-serviced and satisfies the following condition: all of the UAVs that are closer to task j have other tasks that they are closer to than task j. This approach will produce fairly good results in terms of the average waiting time for the tasks. In fact, there is a result for M/G/1 queues, where the service times are known for customers in the queue, which states the minimum processing rule is the policy with the minimum expected waiting time for customers. The minimum processing rule specifies that the server always selects the next customer to be the one with the minimum service time. The M/G/1 result does not hold for the MAV-DTRP consisting of one vehicle since "service times" (travel times between tasks) are dependent. The greedy approach is defined to be analogous to the minimum processing rule of M/G/1 queues.

AVERAGE WAITING TIME Arrival Rate λ = 1/25s, Observance Range = 80km						
Algorithm	**Rep. #1**	**Rep. #2**	**Rep. #3**	**Rep. #4**	**Rep. #5**	**AVG**
Greedy	658.7489	658.6442	646.7785	588.8138	615.4077	*633.6786*
CTS: No penalty T_s=200	551.2836	582.0245	572.1302	523.5565	545.4358	*554.8861*
CTS: T_w=3000 M=100 T_s=200	564.9302	538.3003	579.9729	534.0372	549.7539	*553.3989*
CTS: No penalty T_s=100	538.3433	506.8090	525.0964	531.3029	547.6286	*529.8360*
CTS: T_w=3000 M=100 T_s=100	522.9767	537.0274	561.2702	541.2912	516.3546	*535.7840*
CTS: No penalty T_s=50	531.4476	515.0273	524.0661	500.1980	522.4209	*518.6320*
CTS: T_w=3000 M=100 T_s=50	547.1313	527.0079	519.4846	527.6279	527.3890	*529.7281*
CTS: No penalty T_s=25	581.4458	554.5148	539.7941	537.6909	549.1472	*552.5186*
CTS: T_w=3000 M=100 T_s=25	525.1788	520.4300	549.6089	530.9533	494.5551	*524.1452*

Table 1: Results of the Coordinated Task Scheduling Algorithm

Table 1 shows that the CTS approach gives superior results for the illustrative scenario. Table 1 presents results for the case where the arrival rate is equal to λ = 1/25s and all UAVs have an observance range of 80km. The greedy approach and CTS approach (with different values of T_s and static objective functions) where tested over five different replications of the scenario. The numbers in the table correspond to the average waiting time obtained from the simulation for the associated replication number and approach. Thus, the values under the column "Rep. #1" give the average waiting time for the policies under the first replication. By giving the results in this manner, we can examine how the different approaches performed on exactly the same replication. The results of Table 1 show that the CTS approach significantly outperforms Greedy with respect to the average waiting time measure, with an average improvement of about 15 percent. Note that the average waiting time is effected by the time between selection points T_s. Since this is a dynamic problem it should not be too surprising that the average waiting time is lower for the starving penalty case than for the non-penalty case when T_s=25.

In the no penalty case (static problem is minimizing the average waiting time), some tasks wait an exceedingly long time for service, in excess of 12000s, which is large compared to the average waiting time obtained. By adding a penalty term for tasks that wait more than T_w=3000 seconds for service, the number of starved tasks can be greatly reduced without effecting the average waiting time significantly. Table 2 (3) gives the proportion of tasks that wait more than a prescribed amount of time for service with the Greedy (CTS) approach. The values of Table 3 are representative of all of the CTS approaches in Table 1 that have a penalty term. Tables 2 and 3 give values for the same five replications as listed in Table 1.

GREEDY APPROACH Arrival Rate λ = 1/25s, Observance Range = 80km Proportion of Tasks Waiting More than a Specified Number of Seconds						
TIME	**Rep. #1**	**Rep. #2**	**Rep. #3**	**Rep. #4**	**Rep. #5**	**AVG**
1000	0.1740	0.1637	0.1570	0.1462	0.1520	*0.1586*
1500	0.09929	0.08498	0.08638	0.08364	0.08251	*0.08736*
2000	0.06019	0.06094	0.06032	0.05797	0.05477	*0.05884*
2500	0.04283	0.04673	0.04462	0.04321	0.04042	*0.04356*
3000	0.03138	0.03757	0.03255	0.03058	0.03249	*0.03291*
3500	0.02442	0.02902	0.02658	0.02370	0.02624	*0.02599*
4000	0.01912	0.02189	0.02021	0.01816	0.01940	*0.01976*

Table 2: Greedy Approach to Task Servicing

Looking at the values in Table 3, a CTS approach with a penalty term can significantly reduce the number of tasks that wait more than a given amount of time for service. The CTS approach of Table 3 reduces the proportion of tasks that wait more than 3000s for service. In fact, over the five 16-hr simulation runs, there were no tasks that waited more than 3000s for service. The value of T_w=3000s was not chosen at random but based on previous experimentation, which indicated that T_w should be set so that approximately 3 percent of the tasks wait more than T_w seconds for service with the Greedy approach. Setting T_w this way usually results in the CTS approach, with a starving penalty factor, significantly reducing the proportion of tasks that wait T_w seconds for service without drastically increasing the average waiting time. If the value of T_w is set too low, then the proportion of starved tasks or the average waiting time may be high. If many tasks have waited more than T_w seconds for service (which is likely to happen if T_w is too low), then in an effort to service all the starved tasks first, the vehicles may choose paths that result in a significantly extended waiting time for the non-starved tasks (to the extent that they also become starved in the sense that they wait more T_w seconds for service).

CTS APPROACH WITH T_w=3000, M=100 T_s=50 Arrival Rate λ = 1/25s, Observance Range = 80km Proportion of Tasks Waiting More than a Specified Number of Seconds						
TIME	**Rep. #1**	**Rep. #2**	**Rep. #3**	**Rep. #4**	**Rep. #5**	**AVG**
1000	0.1459	0.1232	0.1270	0.1321	0.1321	*0.1321*
1500	0.0773	0.0646	0.0689	0.0703	0.0692	*0.0701*
2000	0.0480	0.0463	0.0440	0.0504	0.0404	*0.0458*
2500	0.0360	0.0309	0.0280	0.0328	0.0258	*0.0307*
3000	0	0	0	0	0	*0*
3500	0	0	0	0	0	*0*
4000	0	0	0	0	0	*0*

Table 3: Results of the Coordinated Task Scheduling Algorithm with a Penalty

Table 4 gives results in the case where the observance range is 40km; everything else about the experiments is the same as for Table 1, including the replications. As would be expected, the performance of all of the approaches is worse when the observance range is reduced from 80km to 40km. An interesting aspect to note about the results in Table 4 is that the difference between the Greedy approach and the CTS approaches is significantly less than in Table 1. When the observance range is 40km, the CTS approaches only outperform Greedy by about 4 percent as compared to about 15 percent when the observance range is 80km. This may be an indication that the CTS approach performs better, relative to Greedy, if each vehicle has more information about its environment. In general, this seems to be an intuitive notion since future planning becomes more difficult as the uncertainty in the environment increases and the CTS approach maintains a future path plan.

AVERAGE WAITING TIME **Arrival Rate λ = 1/25s, Obs. Range = 40km**						
Algorithm	**1**	**2**	**3**	**4**	**5**	**AVG**
Greedy	691.6044	724.8982	718.2788	695.8353	738.2379	*713.7709*
CTS: T_w=3000 M=100 T_s=100	674.8557	676.4499	690.0139	643.7292	708.2890	*678.6675*
CTS: T_w=3000 M=100 T_s=50	668.9150	694.9385	698.8573	674.2893	711.5369	*689.7074*
CTS: T_w=3000 M=100 T_s=25	722.7441	691.0842	706.9746	644.9454	678.5429	*688.8582*

Table 4: The Measured Effect of a Smaller Observation Range

The results in Table 5 are for an arrival rate of λ = 1/15s; everything else about the experiments is the same as for Table 1. The CTS approach outperformed Greedy by an average of about 11 percent when the arrival rate is 1/15s. In two cases (T_s = 50s and 100s), the average waiting time is better when there was a starving penalty factor (i.e., $M > 0$) as compared to the case where M = 0. The CTS approach with a starving penalty factor (i.e., $M > 0$) also reduced the proportion of tasks waiting more than T_w = 4500 seconds. The value of 4500s was chosen since about 3 percent of the tasks wait that long for service in experiments for the Greedy algorithm and CTS approaches with no starving penalty factor (i.e., M = 0). The CTS approach with T_s = 100 and M = 100 resulted in a proportion of 0.0002 of the tasks waiting more than T_w = 4500s for service, whereas the Greedy approach had a proportion of 0.0318 of the tasks waiting more than T_w = 4500s.

AVERAGE WAITING TIME Arrival Rate λ = 1/15s, Obs. Range = 80km						
Algorithm	**1**	**2**	**3**	**4**	**5**	**AVG**
Greedy	1000.4380	914.3956	983.0897	928.4901	894.1051	*944.1037*
CTS: No penalty T_s=200	853.5781	825.2096	904.6867	824.2652	803.8134	*842.3106*
CTS: T_w=4500 M=100 T_s=200	828.0480	854.2920	894.7123	877.9139	818.3169	*854.6566*
CTS: No penalty T_s=100	839.4062	817.6447	853.4182	851.8693	812.6696	*835.0016*
CTS: T_w=4500 M=100 T_s=100	828.5940	796.7627	857.5051	841.5631	801.5363	*825.1922*
CTS: No penalty T_s=50	842.8012	835.5480	855.1135	825.0411	853.4687	*842.3945*
CTS: T_w=4500 M=100 T_s=50	846.9371	852.5754	843.5708	861.9742	796.5729	*840.3261*
CTS: No penalty T_s=25	861.4406	834.5185	841.7470	837.0753	809.5655	*836.8694*
CTS: T_w=4500 M=100 T_s=25	844.8091	861.3758	829.3821	857.8089	824.2358	*843.5223*

Table 5: Comparison of Results for Several Task Scheduling Algorithms

The experimental results of this section demonstrate the validity of CTS in addressing MAV-DTRP. CTS outperforms the Greedy approach by about 15 percent when the arrival rate is λ = 1/25s and 11 percent when the arrival rate is λ = 1/15s. The performance of CTS is related to the uncertainty each vehicle faces in the environment. For larger communication and observance ranges, the performance of CTS is improved, indicating that CTS is effective when each vehicle has some knowledge about other vehicles.

3.1.4 Discussion of Analytical Results

In this section, the average waiting times (or bounds on the average waiting times) of several control policies are computed analytically and compared with the experimental results obtained in Section 3. All of the policies given in this section are defined specifically for the MAV-DTRP instance of Section 3. The purpose of this is to obtain upper bounds on the average waiting time of an optimal policy for MAV-DTRP and therefore be able to assert with some assuredness that the CTS approach is effective (by comparing its results with the bounds). In other words, it is demonstrated that the experimental results of Section 3 for the CTS approach have superior performance in terms of average waiting time as compared to the policies in this section. The computed average waiting times for the policies in this section serve as a benchmark and provide insight into the effectiveness of CTS.

The paper of Bertsimas and van Ryzin [2] introduce the DTRP and analytically compute the average waiting time for several policies. The results in this paper are applied here to analytically compute the average waiting time of several policies for MAV-DTRP. Recall that for the DTRP the tasks (customers) require an independent and identically distributed amount of on-site service time. Bertsimas and van Ryzin [2] also show that a specified policy is asymptotically optimal in light traffic ($\lambda \bar{s} \to 0$, $\bar{s}$ is the expected on-site service time) and show other policies are within a constant factor of the optimal policy in heavy traffic ($\lambda \bar{s} \to 1$).

The first policy we examine is a first-come first-serve policy with randomized assignments (FCFS-RA). In this policy, the dispatcher randomly assigns tasks of type y_1 to the vehicles. That is, when a task of type y_1 is generated, it is assigned to each vehicle with probability $1/m$ (where m is the number of vehicles that can service that task). A vehicle will only service tasks that have been assigned to it. A vehicle services the tasks that have been assigned to it in a FCFS fashion. Applying FCFS-RA to the scenario of Section 3, which consists of 4 vehicles, a system is obtained that is equivalent to 4 single-server queues, where the arrival rate for each queue is given by $\lambda/4$. From Bertsimas and van Ryzin [2], the average waiting time can be computed for tasks assigned to vehicle i (traveling at a constant rate of 1km/s) with the following formula:

$$W = \frac{\lambda(i) c_2 A}{2\left(1 - \lambda(i) c_1 \sqrt{A}\right)} + c_1 \sqrt{A} \qquad (1)$$

where $c_1 \approx 0.52$, $c_2 = 1/3$, $\lambda(i)$ is the arrival rate for tasks assigned to vehicle i, and A is the area of Ω. The AO Ω must be a square in order for this formula to hold. The term $c_1\sqrt{A}$ is equal to the expected distance between two uniformly and independently distributed points in a square of area A. The term $c_2 A$ corresponds to the expected value of the squared distance between two uniformly and independently distributed points in a square of area A. In the experiments of Section 3, the AO is a 100km × 100km square where the vehicles travel at a rate of 0.156km/s. This set-up is equivalent to the vehicles traveling in a region that is a (100/0.156) km × (100/0.156)km square, where the vehicles travel at a rate of 1km/s. A value of $641.0256^2 \approx (100/0.156)^2$ will be used for the value of A in all of the formulas of this section. Formula (1) for the average waiting time is the well-known Pollaczek-Khinchin (P-K) formula [14]. The original proof of this result assumes that the service times of the queue are independent and identically distributed. In FCFS-RA, the service times, which correspond to the distances between two

tasks, are not independent, but they are identically distributed. Bertsekas and Gallager [13] show that the P-K formula still holds in this situation. The stability condition for FCFS-RA is $\lambda(i)c_1\sqrt{A}<1$. Thus, the stability condition is simply stating that the expected service time must be less than the expected task inter-arrival time. For the scenarios of Section 3, FCFS-RA queue is not stable, which implies that the average waiting time will converge to infinity in the long run. Assuming there are four vehicles, then each vehicle will behave as a single-server queue with a mean service time of 0.52(641.0256) = 333.3333s and a task inter-arrival mean of 60s (if λ = 1/15s) or 100s (if λ = 1/25s). The queue is not stable since the mean service time is greater than the task inter-arrival mean.

There are some policies that are stable for any arrival rate, assuming that they are parameterized correctly. One such approach is a region partitioning scheme. Assume that tasks are randomly assigned to the vehicles in the same manner as described for FCFS-RA. The AO Ω is partitioned into q^2 cells of equal size (imagine taking the square region and slicing it with q rows and $q$columns), where $q > 1$ parameterizes the policy. If q is even, a vehicle can start in one cell, travel through all of the remaining cells and return to the starting cell, where each cell is visited exactly once per cycle (see figure 1a). The formula that is given for the average waiting time of this policy assumes that q is even. Each of the four vehicles will behave independently of each other and only service tasks that have been assigned to it. A vehicle will start in a cell, service all tasks in the cell that have been assigned to it, and then move to the next cell. A vehicle services the tasks in a cell in a FCFS fashion. When a vehicle travels to an adjacent cell, it moves along a straight line to a location that is "projected" from its current location (see figure 1b). This is required to facilitate the mathematical analysis. This policy is referred to as the first-come first-serve policy with randomized assignments and partitioning (FCFS-RA-P).

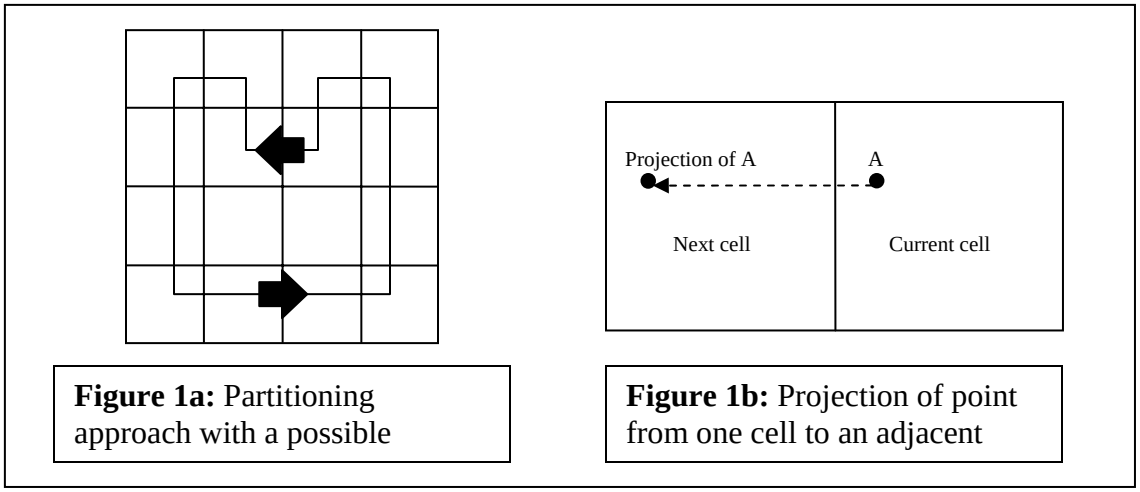

Figure 1a: Partitioning approach with a possible

Figure 1b: Projection of point from one cell to an adjacent

Figure 2: Cell partitioning and UAV movement

For any arrival rate, there exist values of *m* such that the resultant policy is stable. The stability condition for this policy is

$$m > c_1 \lambda(i) \sqrt{A} \,,$$

where $\lambda(i)$ is the arrival rate of tasks being assigned to vehicle *i* and assuming the vehicles travel at a rate of 1km/s. If the stability condition is satisfied and the vehicles travel at a speed of 1km/s, the average waiting time for vehicle *i* is computed with the formula

$$W = \frac{(\lambda(i)/q)c_2(A/q^2)}{2(1-(\lambda(i)/q)c_1(\sqrt{A}/q))} + \frac{1-(\lambda(i)/q^2)c_1(\sqrt{A}/q)}{2(1-(\lambda(i)/q)c_1(\sqrt{A}/q))} q\sqrt{A} + c_1(\sqrt{A}/q)\,.$$

In Table 6, the average waiting time for this policy is given for the arrival rates of Section 3, where *q* is chosen to give the minimal average waiting time. For the scenario of Section 3, note that the queues associated with each of the four vehicles are statistically equivalent and all have the same arrival rate and average waiting time.

Arrival Rate	**Avg. Waiting Time**	"optimal" *q*
1/25s	4315.8929	7
1/15s	7142.8629	11

Table 6: Average Waiting Time for First-Come First-Serve Policy with Randomized Assignments and Partitioning

Another policy that is stable for any arrival rate consists of forming tasks into sets and servicing tasks in each set via an optimal TSP tour. For this policy, an upper bound is given for the average waiting time. Again, tasks are randomly assigned to the vehicles in the same manner as for FCFS-RA and FCFS-RA-P. The vehicles will each operate out of a strategically located depot (center of square region Ω). A vehicle will wait at the depot until *q*tasks have been assigned to it. At that point, the vehicle will service the *q*tasks by following an optimal TSP tour through the tasks. Each subsequent set of *q* tasks will be serviced in the same manner. This policy will be referred to as the TSP-Randomized Assignments (TSP-RA) approach. The stability condition for this policy is that

$$q > (\lambda(i)/4)^2 \beta_{TSP}^2 (641.0256)^2 \,,$$

where $\beta_{TSP} \approx 0.72$ and $\lambda(i)$ is the arrival rate of tasks being assigned to vehicle *i*. When the stability condition is satisfied, an upper bound for the average waiting time W_{TSP} is computed by

$$W_{TSP} \le \frac{(q/\lambda(i))}{2(1-\lambda(i)\beta_{TSP}\sqrt{A/q})} + \frac{q}{2(\lambda(i)/q)} + \beta_{TSP}\sqrt{qA}\,, \qquad \text{see [2].}$$

The upper bound for the average waiting time is given in Table 7, together with the corresponding optimal value of q. In other words, the value of q is chosen to give the least upper bound.

Upper Bound on Average Waiting Time for TSP Approach		
Arrival Rate	**Upper Bound**	"optimal" q
1/25s	4180.5490	26
1/15s	6285.9033	67

Table 7: Upper Bound on the Average Waiting Time for the Traveling Salesman Problem Approach

Comparing the values in Tables 6 and 7 with the results of Section 4, it is apparent that the CTS and Greedy approaches are providing high-quality solutions to MAV-DTRP. The average waiting time for FCFS-RA-P and upper bound for the TSP-RA are approximately eight times larger than the CTS approach. Obviously, an argument could be made that the approaches of this section are naïve or simplistic, but the values in the table do serve as a benchmark and demonstrate that the CTS approach is an acceptable algorithm. The average waiting time for FCFS-RA-P and the upper bound on the average waiting time for TSP-RA cannot be used as an explicit upper bound on the average waiting time for an optimal MAV-DTRP approach since MAV-DTRP is a finite time horizon problem. However, the experiments of Section 3 were run for a significant amount of time (16-hr simulated runs) so that a comparison with the results of this section is reasonable.

3.1.5 Conclusion and Future Directions

Applications that necessitate the need for distributed control are becoming more commonplace and it is now necessary to begin the process of developing and studying techniques that can handle these challenging problems. This paper introduced a problem where the cost of implementing a centralized control approach is prohibitive due to limitations on communication and computational power. A distributed control approach (CTS) was presented that demonstrated superior performance over several baseline approaches. The CTS approach demonstrates that under certain scenarios (such as the scenario of Section 3) it is better to allow for some look-ahead in terms of future planning. Due to the dynamic nature of the MAV-DTRP, it is not simply intuitively obvious that a "future-planning" approach such as CTS can outperform the Greedy approach.

An interesting aspect of MAV-DTRP was discovered through experimental results on an illustrative scenario. The experimental results demonstrate that, in certain situations, the proportion of starved tasks can be reduced without increasing the average waiting time of the control policy. This aspect of MAV-DTRP is analogous to the findings of Powell et al. [3] who show it is better to find sub-optimal solutions of the static optimization problem, in certain situations. If the objective of an MAV-DTRP instance is to minimize the average waiting time, then in some situations it is better to define the static optimization problem to minimize the average waiting time plus a non-zero penalty term rather than just the average waiting time. This paper also introduced a new combinatorial optimization problem GMLP that generalizes the MLP. GMLP is an interesting combinatorial optimization problem that requires further investigation. Since this is a new problem, there is little known about its complexity. From the lemma of Section 2, the problem can be approximated in polynomial time for any class of instances that restrict the penalty functions to be bounded between two linear functions. An interesting open question is whether or not the problem can be approximated in polynomial time, within some factor, for non-linear penalty functions, such as quadratic functions. We conjecture that GMLP is APX-complete for the case consisting of quadratic functions. Another desirable research goal is to develop new algorithms for GMLP and to examine if it is possible to develop an algorithm that performs reasonably well, regardless of the form of the penalty functions. In other words, can the algorithm perform well even when it does not know the form of the penalty functions? It is also desirable to develop algorithms for the particular case where the penalty functions are piecewise linear.

Since this is an emerging research area, the future avenues of research are long and diverse. One possible research direction is to develop other algorithms for MAV-DTRP that introduce other aspects of the problem. For example, the CTS approach (as given in this paper) does not use estimation for vehicle locations that are outside of the observance range to aid in the selection of tasks. A future research direction could be to determine if the performance of CTS can be improved significantly by using estimation for other vehicle locations. Note that, in all likelihood, good estimates of vehicle locations can only be maintained for a short time after vehicles leave the observance range; it is very difficult to estimate the location of a vehicle that has not been in the observance range for a long time. Many aspects to this problem were left out of this paper/research to emphasize the plausibility of distributed approaches to MAV-DTRP. Another direction of future research is to consider more realistic problem formulations by considering task priorities and time windows. In this paper, the problem considered all tasks to

be of equal value; there were no task priorities. A natural extension to the work in this paper is to consider the problem where the tasks have different penalty functions. This paper did consider soft time windows in the sense that there was a penalty for tasks that wait more than a prescribed amount of time for service. Future work could extend this to look at situations where there is also a penalty for servicing a task before a given time. To handle this problem, a penalty function for a task can be defined to be large for times outside of the task's time window.

3.1.6 References

[1] D.J. Bertsimas, G. Van Ryzin, 1993, “Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacitated Vehicles,” *Operations Research* 41(1) 60-76.

[2] D.J. Bertsimas, G. Van Ryzin, 1991, “A Stochastic and Dynamic Vehicle Routing in the Euclidean Plane,” *Operations Research* 39(4) 601-615.

[3] W.B. Powell, M.T. Towns, A. Marar, 2000, "On the Value of Globally Optimal Solutions for Dynamic Routing and Scheduling Problems," *Transportation Science* 34(1) 50-66.

[4] S. Arora, G. Karakostas, 2003, “Approximation Schemes for Minimum Latency Problems,” *SIAM Journal on Computing* 32(5) 1317-1337.

[5] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness (W.H. Freeman and Company, New York, 1979).

[6] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, M. Protasi, Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties (Springer-Verlag, Berlin, 1999).

[7] G. Ghiani, F. Guerriero, G. Laporte, R. Musmanno, 2003, “Real-Time Vehicle Routing: Solution Concepts, Algorithms and Parallel Computing Strategies,” *European Journal of Operational Research* 151(1) 1-11.

[8] L. Bianchi, 2000, “Notes on Dynamic Vehicle Routing – The State of the Art,” *Proceedings of ANTS (Ant Colonies to Artificial Ants) 2000: Second International Workshop on Ant Algorithms* 59-62.

[9] W.B. Powell, W. Snow, R. K. Cheung, 2000, "Adaptive Labeling Algorithms for the Dynamic Assignment Problem," *Transportation Science*, 34(1) 67-85.

[10] D.S. Johnson, L.A. McGeoch, 1997, “The Traveling Salesman Problem: A Case Study in Local Optimization,” in Local Search in Combinatorial Optimization, E.H.L. Aarts, J.K. Lenstra (eds.), John Wiley and Sons, London, UK, 215-310.

[11] K. Moizumi, G. Cybenko, 2001, “The Traveling Agent Problem,” *Mathematics of Control, Signals and Systems* 14(3) 213-232.

[12] P. Kilby, P. Prosser, P. Shaw, 1998, “Dynamic VRPs: A Study of Scenarios,” APES Technical Report APES-06-1998, September 1998

[13] D. Bertsekas, R. Gallager, 1987, Data Networks, Prentice Hall, Englewood Cliffs, N.J.

[14] L. Kleinrock, 1976, Queuing Systems, Vol. 1: Theory, John Wiley, New York.

[15] D.J. Bertsimas, G. Van Ryzin, “Stochastic and dynamic vehicle routing with general demand and inter-arrival time distributions. Advanced Applied Probability, 25:947-978, 1993

[16] Psaraftis, “Dynamic Vehicle Routing: Status and Prospects,” Annals of Operations Research, 61, 143-164, 1995

[17] Gendreau, M., G. Laporte and R. Seguin, 1996a, “Stochastic Vehicle Routing,” Invited Review, European Journal of Operational Research 88, pp. 3-12.

[18] J.D. Papastavrou. “A stochastic and dynamic routing policy using branching processes with state dependent immigration,” European Journal of Operational Research, 95:167-177, 1996.

[19] Minkoff A.S., “A Markov decision model and decomposition heuristic for dynamic vehicle dispatching,” Operations Research 41 77-90, 1993.

[20] D. Bertsimas and D. Simchi-Levi, “A new generation of vehicle routing research: robust algorithms, addressing uncertainty,” Operations Research 44, 1996, 286-304.

[21] Powell, W.B., “A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers,” Transportation Science 30, 195-219 (1996)
[22] M. Gendreau, F. Guertin, J.Y. Potvin, R. Seguin, “Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries,” technical report CRT-98-10, Center of Transportation Research, University of Montreal, 1998
[23] H. N. Psaraftis, “Dynamic vehicle routing problems,” in Vehicle Routing: Methods and Studies, B.L. Golden and A.A. Assad (eds.), 223-248, Elsevier Science Publishers, 1988.
[24] A.C. Regan, J. Herrmann, and X. Lu, “The Relative Performance of Heuristics for the Dynamic Traveling Salesman Problem,” proceedings of the 81st meeting of the Transportation Research Board, 2002.
[25] S. Ichoua, M. Gendreau, J.Y. Potvin, “Exploiting Knowledge about Future Demands for Real-Time Vehicle Dispatching,” Centre de Recherche sur les Transport, Univerity of Montreal, 2000.

4 The TASK Project Wide Demonstration

At the direction of the DARPA Program Manager, AIT facilitated a project-wide demonstration at which all TASK Principal Investigators made available demonstrations of the results of their research efforts supported by the TASK program. The demonstration took place August 4, 2004 in the ballroom of the Hotel Washington, Washington, DC. The DARPA Program manager invited potential guests. The following table lists those who attended

.

Last Name	First Name	Organization	Email
Cobble	Kevin	L-3 Communications Integrated Systems	Kevin.L.Cobble@L-3com.com
DiPierro	Gus	JTRS JPO	gus.dipierro@hqda.army.mil
Frazier	Tiffany	ALPHATECH	tiff@dc.alphatech.com
Jones	Harold	ALPHATECH	hjones@dc.alphatech.com
Lala	Jaynarayan	Raytheon Company	Jay_Lala@raytehon.com
Moore	Bill	US ARMY INTELLIGENCE CENTER	WILLIAM.MOORE@HUA.ARMY.MIL
Nagy	James	AFRL/IF	James.Nagy@rl.af.mil
Naqvi	Waseem	Raytheon	Waseem_Naqvi@raytheon.com
Rzepka	Bill	AFRL - Rome	rzepkaw@rl.af.mil
Schoenwald	Josh	The Ohio State University / Army Research Laboratory	iokua@mac.com
Swan	Stephen	TSM FCS	steve.swan@hqda.army.mil
Tittle	James	Ohio State University / Dept. of Industrial Eng	tittle.2@osu.edu
Trent	Stoney	The Ohio State University / Army Research Laboratory	stoney.trent@us.army.mil
Veney	David	PM Soldier Warrior	David.Veney@US.Army.Mil
Voshell	Martin	C/S/E/L Ohio State Univ.	mvoshell@mac.com
Wagner	Tom	DARPA / IPTO	twagner@darpa.mil
Waszak	Martin	NASA Langley Research Center	m.r.waszak@larc.nasa.gov
Zachery	Randy	Army Research Office	randy.zachery@us.army.mil
Zwan	Allen	BAE Systems	allen.zwan@baesystems.com

Table 8: Attendees of the TASK Project-wide Demonstration

4.1 Organization of the Demonstration

Through discussions at an OEF Demonstration Workshop held at the University of Texas at Austin, July 1-2, 2003 and numerous teleconferences, all facilitated by AIT , the Principal Investigators agreed that, at a minimum, each demonstration would show how the Principal Investigator's research might handle a specific set of events in a larger military operation scenario based upon the OEF. A *narrative integration*, which consisted of the timeline annotated with events and a description of the military operations, provided continuity between the individual demonstrations. When they entered the demonstration, attendees received a handout of the narrative integration, and could use it sample the individual organizations' demonstrations in any order.

The TASK Demonstration had purely software-based as well as hardware-based demonstrations. The researchers presenting hardware-based demonstration used a variety of robotic platforms, and, in general, moved what were purely software-based research results to autonomous, cooperative robotic platforms. Several groups offered both hardware and software demonstrations. The following table contains a list of the Principal Investigators' organizations and the research area demonstrated.

Organization	Research Area	WWW Address
ALPHATECH Inc.	Coordination in Multi-Agent Systems	http://www.alphatech.com
Dartmouth College	Agent-Based Systems Engineering	http://actcomm.thayer.dartmouth.edu/task/
Hampshire College	Multi-type, Self-adaptive Genetic Programming for Complex Applications	http://hampshire.edu/lspector/darpa-selfadapt.html
Metron, Inc.	Agent-Based Computing	http://www.metsci.com/abc/
Massachusetts Institute of Technology/BBN	Creation, Modeling, and Analysis of Adaptive Agent Systems	http://omar.bbn.com/MIT/AdaptiveAgents/
Santa Fe Institute	Dynamics of Learning	http://www.santafe.edu/~dynlearn/
Stanford University	Control and Coordination with Game Theoretic Agents	http://task.stanford.edu/
University of Illinois at Urbana-Champaign	A Parametric Model for Large-Scale Agent Systems	http://osl.cs.uiuc.edu/
University of Massachusetts	Analytical Tools for Agent-Based Computing	http://kdl.cs.umass.edu/
University of New Mexico	Computation in the Wild: Moving Beyond the Metaphor	http://www.cs.unm.edu/~immsec/
University of Southern California Information Sciences Institute	Mathematical Modeling of Multi-Agent Systems	http://www.isi.edu/~lerman/projects/task/
University of Texas at Austin	Multi-Scale Behavioral Modeling and Analysis Promoting a Fundamental Understanding of Agent-Based System Design and Operation	http://www.lips.utexas.edu/UTAustin/AgentDesign/

Table 9: TASK Demonstration Organizations and Research Areas

4.2 The Military Scenario for the TASK OEF Demonstration

The military scenario was a simple extension of the problems of the TASK OEF. The mission consists of a set of medium and small UAV sensor platforms and networks of Unattended Ground Sensors (UGSs) that perform surveillance and reconnaissance services on stationary and moving ground targets performing surveillance and reconnaissance services on stationary and moving ground targets. For the TASK OEF demonstration, the purpose was for the Principal Investigators to demonstrate the UAVs and UGSs operating as autonomous entities in a multi-agent system and handling the situational awareness activities through their approaches to cooperation and adaptation. The mission has a definite beginning and end and has a finite duration. The multi-agent system of UAVs and UGSs must maintain specified levels of accuracy for target identification, location, and velocities within sub-areas of the area of operations, and the information for a sub-area must be no older than a specified age. The accuracy requirements, age limits, and sub-area boundaries change as the mission evolves. The location and identification requirements have high accuracy since the area of operations includes both hostile and non-hostile targets. Data and information fusion, if needed, were to be mocked-up. AIT filled out this skeletal mission scenario for those Principal Investigators who required more details.

Organization	Demo Participants	Other
ALPHATECH	Larry Roszman	Angell Rosa
	Derek Armstrong	Tiffany Frazier
	Aram Khalili	Hal Jones
	Gwen Hickling	Chuck Morefield
	Danny Russell	
BBN/MIT	Oliver Selfridge	
	Brett Benyo	
	David Montana	
	Wally Feurzeig	
Dartmouth	George Cybenko	
	Alex Jordan	
	Wayne Chung	
	Valentine Crespi	
	Daniela Rus	
Hampshire	Lee Spector	
Metron	Greg Godfrey	Tom Mifflin
	Aren Knutsen	
	John Cunningham	
SFI	Jim Crutchfield	
	Michael Schippling	
Stanford	Ryan W. Porter	
	Yoav Shoham	
UIUC	Gul Agha	
	Tom Brown	
	MyungJoo Ham	
UMass	David Jensen	
	Andrew Fast	
UNM	Hajime (Jim) Inoue	
	Stephanie Forrest	
USC ISI	Kristina Lerman	
UTAustin	Suzanne Barber	
	Tom Graser	
	David Han	
	Dung Lam	

Table 10: Participants in the TASK OEF Demonstration

4.2.1 The Area of Operations

The area of operations is stationary and well-defined with mixed terrain. A river valley runs north-south with feeder streams flowing from high mountains to the north and west. The valley has low mountains to the east and south, which slope to plains. The valley has farming and grazing with pockets of forest. The mountain slopes are forested.

Villages are scattered throughout valley, and a medium sized city occupies its southeast end. Well-maintained, hard-topped roads connect the villages and city. Jeep and walking trails extend into both high and low mountains. Civilian automobiles, trucks, busses, farm equipment, etc.

travel the roads. The community has a modern infrastructure, which includes an electric grid, distributed natural gas, wired and cellular telephone, buried fiber-optics, and the Internet. Both the city and the villages have small manufacturing factories.

Outside hostile forces have occupied parts of the valley. The friendly forces are not native. Several villages are sympathetic to occupying hostile forces, and the remainder of population is friendly to neutral toward friendly forces. The hostile forces' supply lines traverse the western high mountains and northeastern low mountains.

Within the area of operations, specific sub-areas may be known by the friendly forces to contain possibly potential hostile targets, but the specific existence, location, and identification of actual targets is unknown. Some sub-areas may be known to present a high potential threat level to surveillance UAVs, although specific threats such as ground-to-air missiles are unknown. The occurrence and definition of sub-areas that present threats and potential reconnaissance and surveillance targets will change throughout the mission. The base-station communicates these changes to the multi-agent system of UAVs as communications permit. In addition, threats to UAVs may appear suddenly – popup – in sub-areas assumed to have a low threat level.

4.2.2 Reconnaissance and Surveillance Targets

The reconnaissance and surveillance targets – known and unknown, stationary and moving – will appear and disappear throughout the mission. A base-station, which represents the central command node, communicates to the UAV multi-agent system the locations of known targets and regions that may contain potential hostile targets and receives the surveillance and reconnaissance data from the multi-agent system of UAVs as communications permit. The targets can present the following problems:

1. Some targets have accurately known locations but must be identified to determine to type and possible hostile nature
2. Some targets' locations are inaccurate or erroneous, must be determined accurately, and must be identified as well
3. Some targets require simultaneous location-identification-velocity surveillance by two different UAVs: for example, stereoscopic SPOT Synthetic Aperture Radar (SAR) or triangulating MTI
4. Some targets are *time critical*, that is, they must be serviced within a particular time window

Moving targets can present the additional problems.

1. Targets, initially stationary, may move, and targets, initially moving, may stop moving
2. The identification of moving targets must be sufficiently to differentiate hostile from non-hostile, such as a farm truck from a military truck
3. Moving targets may attempt to evade surveillance

4.2.3 Communications

Inter-UAV communications are both a necessity and a limitation for coordinated and adaptive surveillance and reconnaissance by a multi-agent UAV systm of popup targets, which can be hostile. AIT introduced a few limitations and potential problems into the military scenario's communications description. First, all a UAV's communicates with other UAVs, sensors, and

base-stations are only line-of-sight and can occur within only a maximum radius. Due to geo-spatial topology, atmospheric conditions, jamming, higher priority messaging, and a technologically sophisticated hostile force, the following events may occur:

1. The maximum communications distance for a UAV may change unexpectedly
2. All inter-UAV and base-station communications may be lost completely for finite durations
3. Any individual message may be garbled, partially lost, or totally lost
4. The maximum bandwidth available to a UAV may decrease or increase unexpectedly
5. False messages may be transmitted by the hostile forces
6. Messages may be intercepted by hostile forces.

4.2.4 Unattended Ground Sensor Networks

AIT added Unattended Ground Sensor (UGS) Networks to the scenario to provide an additional dimension that mirrors one of the new directions of military intelligence data gathering, to introduce a multi-agent system that operates on a smaller computational scale that than represented by the UAV multi-agent system, and to introduce cooperative interactions between multi-agent systems. The UGSs in this scenario have low-powered transceivers similar to the Mica Motes of the DARPA NEST project. UGS networks may be spread across some sub-areas. Some UAVs are equipped to communicate with the UGS networks. The geo-location of some the sensors may be known, and the UAVs can assist with some of the sensors that have unknown locations determine their approximate locations if TASK coordination research results are demonstrated.

As with all communications, communications between nodes of an UGS network and a UAV may be disrupted by weather or jamming. Due to either equipment failure or hostile action, individual sensors may disappear from an UGS network. Some individual sensors in a network may be mobile over small distances. In general, such movement is for purpose of establishing or maintaining the situational awareness accuracy.

4.2.5 The Scenario Timeline

Since several of the Principal Investigators research focused on the design of multi-agent systems to meet specific task performance demands and others focused on evaluation of how effectively the multi-agent system performed, AIT placed both pre- and post- mission execution periods in the timeline. During the pre-execution period, the Task Force Commander states the details of the mission. This statement includes communication to the UAVs the following:

1. The planned duration of the mission
2. Lists of accuracies and lifetimes to be maintained for the situational awareness of the various sub-areas of the area of operations
3. Lists of known ground sites that require surveillance
4. Lists of sub-areas that may contain unknown targets
5. Lists sub-areas that may contain threats
6. Lists sub-areas that are known to contain threats.

With this information the multi-agent system designers or, if possible, the UAV multi-agent system itself, develop the initial system-wide plan to accomplish the mission. Then the UAVs disperse begin their tasks. Those UAVs that interact with the UGS networks assist their networks in geo-spatial location of individual nodes if necessary.

During mission execution the UAV multi-agent system, performs a variety of normal activities.

1. A UAV performing surveillance of a known ground broadcasts the data it gathered to all UAVs with which it can communicate and, if possible, to the base-station
2. A UAV, finding a previously unknown target while performing reconnaissance, broadcasts the discovery, within communications limits, to all UAVs and the base-station
3. The base-station broadcasts the locations of new known location targets to all UAVs as soon they become available.
4. UAVs coordinate within the multi-agent system to perform surveillance of targets that require multiple observations, that is, cross-mission tasking targets
5. The UAVs coordinate within the multi-agent system to maintain the required accuracy and lifetime requirements for all sub-areas within the area of operations.

4.2.6 Disruptive Events

Besides the appearance and disappearance of possible targets and hostile areas and changing accuracy requirements, the UAV and UGS multi-agent systems must adapt to unpredictable disruptive events. These disruptive events for the TASK OEF demonstration focused on communications and equipment degradation and failure.

Because of jamming, communications between all or select UAVs, UAVs and the base-station, and UAVs and UGS networks can be disrupted. During a disruptive event, the UAVs should continue to perform their usual surveillance, reconnaissance, and tracking tasks. Each UAV or UGS stores data as well as its capabilities permit, and in accordance with its contribution to the situational awareness accuracy and lifetime requirements. Once the jamming ceases and all communications are restored, UAVs broadcast their stored data, retrieve and process data from the UGS networks, and coordinate with in multi-agent system to evaluate the current situational awareness accuracies and lifetimes and to determine re-tasking. A major coordination problem for the UAV multi-agent system to effectively coordinate immediately after communications restoration when the communications channels may be so overwhelmed that data and information may be lost. Similar communications disruption problems occur during storms and when landscape such as mountains, trees or buildings impede lines-of-sight.

A UAV may leave the multi-agent system because of hostile action or equipment failure; in which case, the UAVs in the system must coordinate an adaptation that is appropriate to maintaining the required situational awareness accuracies and lifetime limits. New UAVs joining the multi-agent system necessitate similar activities. Sensors on the UAVs or UGSs may stop working, in a more difficult case, fail or degrade with or without obvious symptoms.

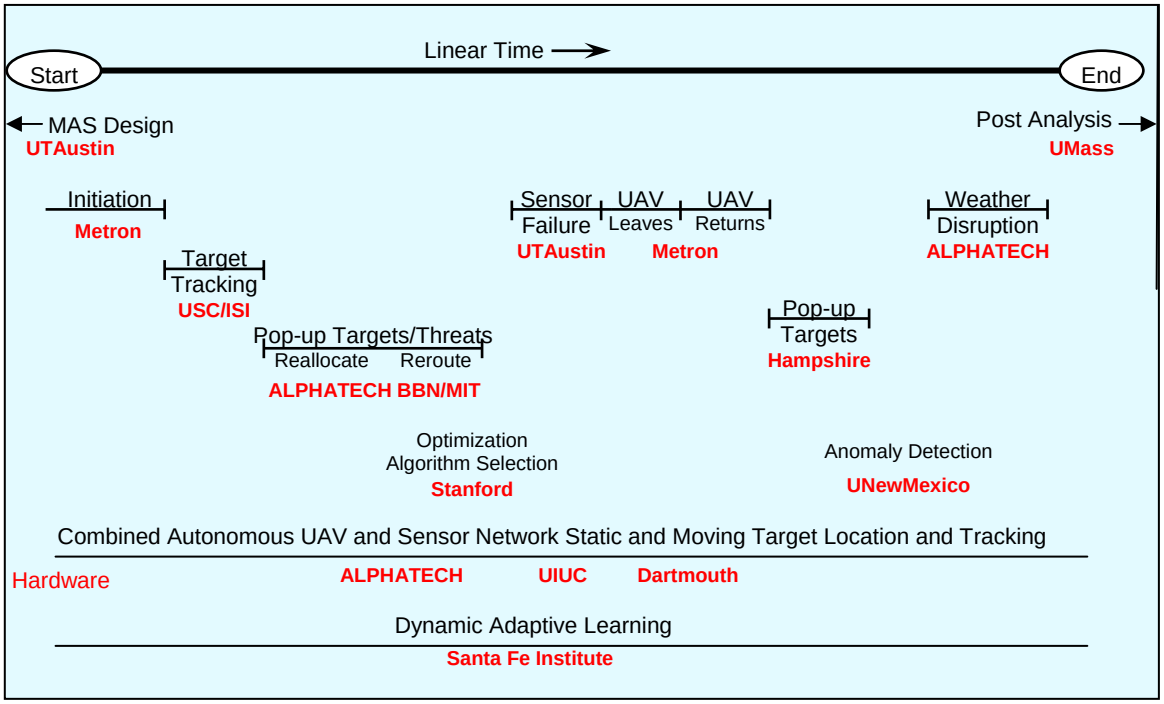

Figure 3: Timeline Focus and Event Focus of the Principal Investigators' Demonstrations

5 The BAE SYSTEMS Advanced Information Technologies Demonstration

AIT (formerly ALPHATECH, INC) presented two demonstrations at the project-wide TASK OEF Demonstration in August 2004. One demonstrated the algorithmic details of the research on minimizing unsurveyed target's lifetimes discussed in Section 2. This software-based demonstration employed AIT's Testbed for Taskable Agent Systems (TTAS), developed under the TASK program, to demonstrate the algorithms in operation. In the second AIT implemented and demonstrated these same algorithms on real robotic systems. To this hardware-based demonstration, AIT added a sensor network composed of MICA2 Motes, developed under the DARPA NEST program, that collaboratively localized target locations and communicated the localization to the robots. This section contains a description of the second demonstration.

5.1 Overview of the AIT Hardware-based Demonstration

The purposes of the research effort behind AIT's hardware-based demonstration was first to investigate the performance of Coordinated Task Scheduling (CTS) algorithm, discussed in Section 2, within a multi-agent system of autonomous, coordinating robots and second to examine implementing a complex algorithm such as CTS in a realistic hardware environment. Because of time constraints, AIT confined its hardware-based investigations to the **Stationary Targets** and **Cross-mission Tasking** surveillance problems of the OEF – problems **UAV-S (1)** and **UAV-S (2)** – which fit easily within the demonstration scenario.

The robotic multi-agent system used by AIT was composed of three ActivMedia Pioneer 2 DXe robots, which acted as stand-ins for the UAVs. The robots came equipped as follows:

- On-board Pentium III 800 Mhz computer, 256 Kbytes RAM, 10 Gig Harddrive
- 802.11b wireless Ethernet communications
- SICK laser scanning, range measurements
- On-board video camera
- Frame grabber with software recognizes color and can navigate with respect to color
- Forward looking sonar
- Inertial components, compass/inclinometer to supplement wheel odometer and direction readings
- The Linux operating system
- ARIA: a low-level multi-threaded robotics server and software library that provides summary navigation commands, obstacle avoidance, health and safety, Ethernet communications, map navigation, video camera and frame grabber control, etc.
- Additional ActivMedia software for automated robot localization and navigation through sonar and laser readings and map comparisons

AIT implemented the CTS algorithm in the native C++ of the ActivMedia ARIA libraries and integrated it with the ARIA libraries and other ActivMedia software. By using the ActivMedia libraries and software for robot localization, navigation between waypoints, obstacle avoidance, camera control, and communications, AIT concentrated its effort on the essential TASK related research efforts.

The demonstration scenario for the ActivMedia robotic multi-agent system was the following:

1. Establish and map a physical environment with navigation reference points

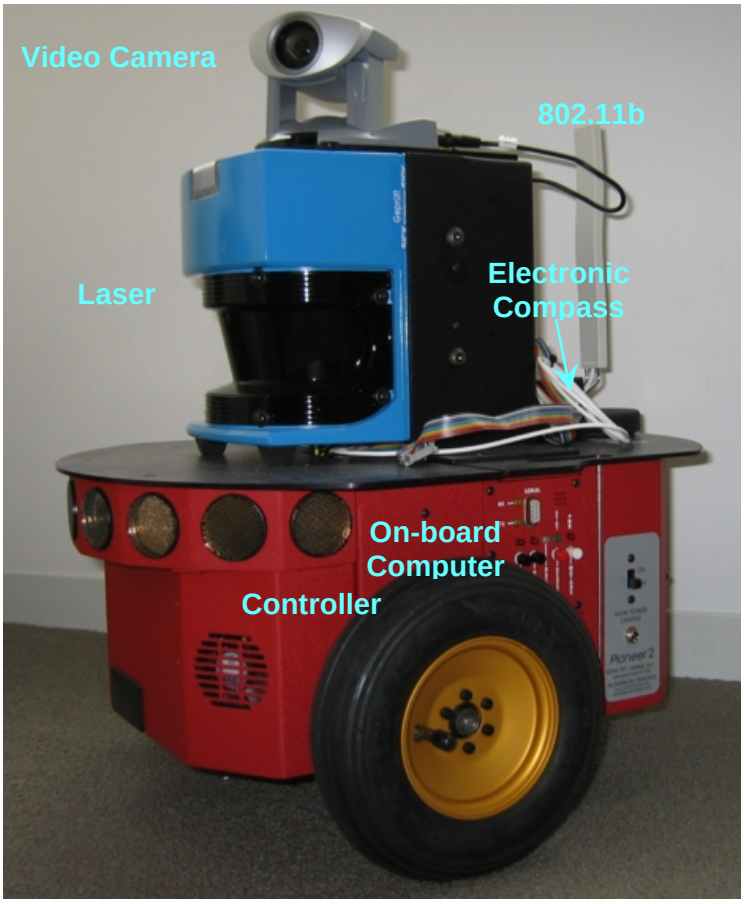

Figure 4: ActivMedia Pioneer 2-DXe

2. Add obstacles
3. Place targets at known locations, some of which require cross-mission tasking, that is, two robots must observe the target simultaneously
4. Occasionally have the base-station announce via 802.11 the target locations to the robots, who apply the CTS algorithm to determine navigation waypoints
5. The robot navigates towards the first waypoint indicated by the CTS algorithm and avoids obstacles that its laser and sonar readings indicate are in its path
6. When a robot, through its localization algorithms, decides it is near a target, use the video camera and the color tracking software to complete moving to the target
7. Fill the video frame with the target, and transmit a video of the target to the base-station
8. Occasionally add new targets and repeat the announcements
9. When a robot encounters a cross-mission task, coordinate with another robot to arrive at the target from two different directions and provide simultaneous video of the target

Each MICA2 Mote[1], twenty of which form the UGS network for the AIT demonstration, is composed of two connected boards. One board holds a microprocessor, FM transceiver, AD converter, UART, parallel connector for programming, serial connector for data transfer, and holder for two AA batteries. The other holds sensors and plugs into the first by a 51 pin connector. The microprocessor has the following characteristics:

- Atmel ATmega 128L
- 512K bytes flash memory for programs
- 4K bytes EEPROM; AD 10 bit, 8 channel
- Other interfaces DIO, I2C, SPI
- Current draw 8mA active mode , < 15 μA sleep mode

The transceiver has the following characteristics:

- ChipCon CC1000
- Center Frequency 868 or 916 MHz
- Number of Channels > 4, > 50 programmable and country specific
- Data Rate 38.4 Kbaud
- RF Power -20 to +5 dBm programmable
- Receiver sensitivity -98 dBm
- Outdoor range 500 ft line-line-of-sight with external 1/4 wave dipole antenna
- Current draw 27mA at maximum transmit power, 10 mA receive, < 1 μA sleep

The system can use many different sensors. The particular sensor board that AIT used has the following sensors:

- Photocell
- Thermistor
- Microphone
- 2-Axis Magnetometer
- 2-Axis Accelerometer

Experiments demonstrated that the microphone sensor and an acoustic target were most effective for the TASK OEF demonstration.

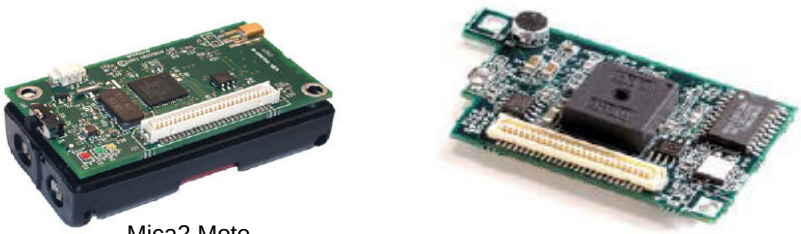

Figure 5: MICA2 Mote and Sensor Board

A project supported by the DARPA NEST program and Intel at the University of California Berkeley[2] developed a programming language *nesC*, which is an extension to the C programming language, and a real-time, event driven operating system *TinyOS* for the MICA2

[1] AIT purchased the MICA2 Motes from Crossbow, Inc., http://www.xbow.com.
[2] Details are available at http://webs.cs.berkeley.edu/.

Motes. The language and operating system have libraries that control all the Mote hardware and provide related high-level application programming interfaces to application designers and programmers. AIT designed and implemented software agents that reside on the Motes and coordinate to localize acoustic targets.

The demonstration scenario for the MICA2 Motes UGS network was as follows:

1. Place the Motes at known locations within the physical environment
2. Send a message to each Motes that tells it its location
3. Broadcast the command to all Motes to determine that they are to find their four nearest neighbors
4. Sound an acoustic target within the physical space covered by the UGS network
5. The Motes who heard the target exchange detection and non-detection messages with their neighbors
6. Each Motes uses a geometric algorithm to compute the geometric block in which the acoustic target is located and refines the computation by message exchanges with its neighbors
7. The block location of the target is broadcast throughout the network

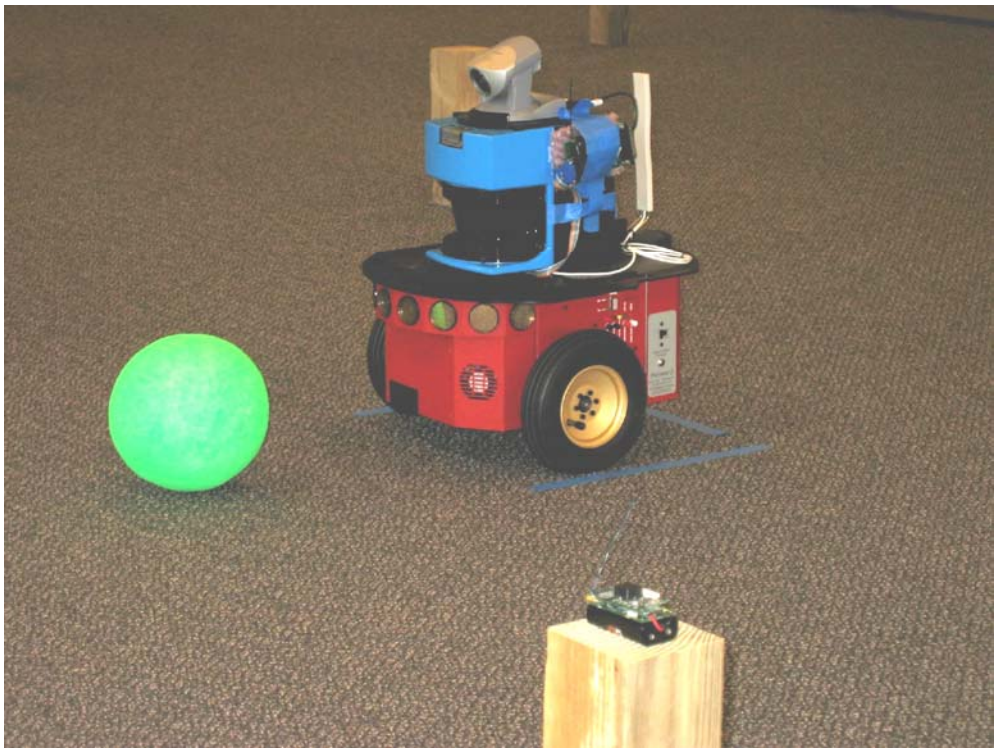

Figure 6: A robot performing video surveillance of a target with an acoustic UGS in the foreground.

To enable the ActivMedia robot multi-agent system and the MICA2 Mote UGS network, AIT integrated a Mica2 Mote transceiver-microprocessor board with each robot's on-board computer and designed and implemented software that allows the two to communicate. The two multi-agent systems communicate through this interface. The following steps complete the demonstration scenario:

8. A robot queries the UGS network for possible acoustic targets locations

9. The robot passes the coordinates of a corner of the target locale block to its on-board CTS algorithm, which computes a new set of waypoint that are optimized to minimize unserveyed target lifetimes
10. The robot navigates to the target locale and uses the ActivMedia color tracking software to locate the target.